Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques

APPROX/RANDOM 2026, August 19–21, 2026,
Boston University, Boston, Massachusetts, USA

Edited by

Mohit Singh

Tom Gur



Editors

Mohit Singh
Georgia Institute of Technology, Atlanta, GA, USA
mohit.singh@isye.gatech.edu

Tom Gur
University of Cambridge, UK
tom.gur@cl.cam.ac.uk

ACM Classification 2012
Theory of computation

ISBN 978-3-95977-449-9

Published online and open access by
Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at https://www.dagstuhl.de/dagpub/978-3-95977-449-9.

Publication date
September, 2026

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists all publications of this volume in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at https://portal.dnb.de.

License
This work is licensed under a Creative Commons Attribution 4.0 International license (CC-BY 4.0): https://creativecommons.org/licenses/by/4.0/legalcode.
In brief, this license authorizes each and everybody to share (to copy, distribute and transmit) the work under the following conditions, without impairing or restricting the authors' moral rights:
- Attribution: The work must be attributed to its authors.

The copyright is retained by the corresponding authors.

Digital Object Identifier: 10.4230/LIPIcs.APPROX/RANDOM.2026.0

ISBN 978-3-95977-449-9 **ISSN 1868-8969** **https://www.dagstuhl.de/lipics**

LIPIcs – Leibniz International Proceedings in Informatics

LIPIcs is a series of high-quality conference proceedings across all fields in informatics. LIPIcs volumes are published according to the principle of Open Access, i.e., they are available online and free of charge.

Editorial Board

- Patricia Bouyer-Decitre (Université Paris-Saclay, CNRS, ENS Paris-Saclay, FR)
- Roberto Di Cosmo (Inria and Université Paris Cité, FR)
- Faith Ellen (University of Toronto, CA)
- Holger Hermanns (Universität des Saarlandes, Saarbrücken, DE and Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Wadern, DE)
- Laura Kovács (TU Wien, AT)
- Daniel Král' (Leipzig University, DE and Max Planck Institute for Mathematics in the Sciences, Leipzig, DE – Chair)
- Sławomir Lasota (University of Warsaw, PL)
- Meena Mahajan (Institute of Mathematical Sciences, Chennai, IN)
- Chih-Hao Luke Ong (Nanyang Technological University, SG)
- Eva Rotenberg (IT University of Copenhagen, DK)
- Pierre Senellart (ENS, Université PSL, Paris, France)
- Alexandra Silva (Cornell University, Ithaca, US)

ISSN 1868-8969

https://www.dagstuhl.de/lipics

Contents

Preface
Mohit Singh and Tom Gur 0:xi

Program Committees
..... 0:xiii

Subreviewers
..... 0:xv

List of Authors
..... 0:xxi

APPROX

Improved Approximation Algorithm for Maximum Balanced Biclique
Pasin Manurangsi 1:1–1:9

Stochastic Caching via Subset Entropy
Ravi Kumar, Roie Levin, Joseph (Seffi) Naor, and Debmalya Panigrahi 2:1–2:24

Socially Fair Clustering: Parameterized Approximation and Local Search
Aditya Anand, Yury Makarychev, and Liren Shan 3:1–3:22

Near-Optimal Streaming Approximation for Max-DICUT in Sublinear Space Using Two Passes
Santhoshini Velusamy 4:1–4:23

Incremental Dominating Set
Ilan Doron Arad, Jonathan Gal, and Joseph (Seffi) Naor 5:1–5:21

Bichromatic Geometric Spanners
Theodore Fung and Csaba D. Tóth 6:1–6:16

Improved Approximation Algorithms for Bounded-Length Path Augmentation
Felix Hommelsheim 7:1–7:22

Capacitated Partition Vertex Cover and Partition Edge Cover
Rajni Dabas, Samir Khuller, and Emilie Rivkin 8:1–8:24

A Bad Example for Jain's Iterative Rounding Theorem for the Cover Small Cuts Problem
Miles Simmons, Ishan Bansal, and Joe Cheriyan 9:1–9:15

Hardness of the Binary Covering Radius Problem in Large ℓ_p Norms
Huck Bennett and Peter Ly 10:1–10:21

Improved Algorithms and Lower Bounds for Parametrized Metrical Service Systems
Junhao Gan, Xiao Sun, and Seeun William Umboh 11:1–11:18

Uncrossed Multiflows and Applications to Disjoint Paths
Chandra Chekuri, Guyslain Naves, Joseph Poremba, and F. Bruce Shepherd 12:1–12:23

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2026).
Editors: Mohit Singh and Tom Gur
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Approximating (Weighted) Chromatic Correlation Clustering via Cluster LP
Fateme Abbasi, Hyung-Chan An, Jarosław Byrka, Changyeol Lee, and Yongho Shin 13:1–13:20

Hitting Axis-Parallel Segments with Weighted Points
Rajiv Raman, Siddhartha Sarkar, and Jatin Yadav 14:1–14:22

Length-Constrained Network Design in Planar Digraphs
Chandra Chekuri and Rhea Jain 15:1–15:24

Directed Low Diameter Decomposition for Structured Digraphs
Shinwoo An and Arnold Filtser 16:1–16:24

Bi-Lipschitz Extensions and Outlier Embeddings into Trees
Shuchi Chawla, Arnold Filtser, Kristin Sheridan, and Yonatan Trachtenberg 17:1–17:24

Tight Approximation Results for Matroid Optimization with a Linear Constraint
Ilan Doron-Arad, Hadas Shachnai, and Gilad Shmerler 18:1–18:19

Strong Inapproximability for a Promise Rank Problem
Venkatesan Guruswami, Xuandi Ren, and Shaoxuan Tang 19:1–19:22

On the Approximability of Parameterized Minimum Monotone Satisfying Assignment
Venkatesan Guruswami, Bingkai Lin, Xuandi Ren, and Xin Zheng 20:1–20:14

Cycle Cancellation for Submodular Fractional Allocations and Applications
Chandra Chekuri, Pooja Kulkarni, Ruta Mehta, and Jan Vondràk 21:1–21:20

A Configuration-LP Framework for Connected k-Median Clustering
Kushagra Chatterjee, Rojin Rezvan, and Ali Vakilian 22:1–22:20

Approximation Algorithms for Matroidal Prerequisite Systems
Robert P. Streit and Vijay K. Garg 23:1–23:24

The Code Distortion Problem
Huck Bennett, Matthew Fox, and Bryant Morrell 24:1–24:22

Online Matching with Size-Based and Convex Delays
Junhao Gan, Xiao Sun, and Seeun William Umboh 25:1–25:23

Online TCP Acknowledgment Under General Delays
Sujoy Bhore, Michał Pawłowski, and Seeun William Umboh 26:1–26:24

Approximation Algorithms for Discounted Graph Search with Norm Objectives
Svenja M. Griesbach, Felix Hommelsheim, and Max Klimm 27:1–27:20

Threshold Rounding and Bounded-Degree Boolean MAX 2-CSP
Suprovat Ghoshal, Neng Huang, Euiwoong Lee, Konstantin Makarychev, and Yury Makarychev 28:1–28:21

Incremental Consistent k-Center Clustering
Mara Grilnberger and Antonis Skarlatos 29:1–29:19

Optimal Stable Coresets for Geometric Median via Uniform Sampling
Amir Carmel, Robert Krauthgamer, and Nir Petruschka 30:1–30:14

RANDOM

Unbounded-Width CSPs Are Untestable in a Sublinear Number of Queries
Yumou Fei 31:1–31:9

Tight Simulation of a Distribution Using Conditional Samples
Tomer Adar 32:1–32:20

When Local and Non-Local Meet: Quadratic Improvement for Edge Estimation with Independent Set Queries
Tomer Adar, Yahel Hotam, and Amit Levi 33:1–33:21

Two-Sided Lossless Expanders in the Unbalanced Setting
Eshan Chattopadhyay, Mohit Gurumukhani, Noam Ringach, and Yunya Zhao 34:1–34:19

Towards Tight Bounds for Testing k-Colorability
Nick Kushnir and Asaf Shapira 35:1–35:13

Distributed Gaussian Mean Testing Under Communication Constraints: Messages, Samples, and Coins
Clément L. Canonne and Nimitt 36:1–36:20

Hard-To-Sample Distributions from Robust Extractors
Farzan Byramji, Daniel M. Kane, Jackson Morris, and Anthony Ostuni 37:1–37:22

Almost All Graphs Are Vertex-Minor Universal
Ruben Ascoli, Bryce Frederickson, Sarah Frederickson, Caleb McFarland, and Logan Post 38:1–38:20

Improved Error Reduction for Weighted PRGs
Ben Chen, Gil Cohen, Dean Doron, Yuval Khaskelberg, and Amnon Ta-Shma 39:1–39:23

Output-Sparse Matrix Multiplication Using Compressed Sensing
Huck Bennett, Karthik Gajulapalli, Alexander Golovnev, and Evelyn Warton 40:1–40:20

From Decision to Random Certificates: Exponential Separation for Edge Estimation with Independent Set Queries
Debarshi Chanda, Buddha Dev Das, Arijit Ghosh, and Gopinath Mishra 41:1–41:23

Glauber Dynamics for Random Field Ising Models on Bounded Degree Graphs
Yi Han 42:1–42:18

Markov Chains with Rewinding
Amir Azarmehr, Soheil Behnezhad, Alma Ghafari, and Madhu Sudan 43:1–43:24

One-Way Functions and Polynomial-Time Dimension
Satyadev Nandakumar, Subin Pulari, Akhil S, and Suronjona Sarma 44:1–44:23

Recovering Planted Colorings in Sublinear Time
Weronika Wrzos-Kaminska 45:1–45:23

One-Sided-Error Parameterized Reductions for the Minimum Distance and Shortest Vector Problems
Shuichi Hirahara and Kazuki Ogitsuka 46:1–46:22

Sequential Sweeps and High Dimensional Expansion
Vedat Levi Alev and Ori Parzanchevski 47:1–47:24

An Elementary Proof of the First LP Bound on the Rate of Binary Codes
Nati Linial and Elyassaf Loyfer 48:1–48:10

Ultra-Sparse Expanders and the Free Method
Gil Cohen and Gal Maor 49:1–49:20

Entropy Equivalence Testing
Clément L. Canonne, Yash Pote, Jonathan Scarlett, and Joy Qiping Yang 50:1–50:23

On Computing Total Variation Distance Between Mixtures of Product Distributions
Weiming Feng, Yucheng Fu, Minji Yang, and Anqi Zhang 51:1–51:21

Testing k-Submodularity
Themistoklis Haris and Diptaksho Palit 52:1–52:21

Symmetric Distributions from Shallow Circuits
Daniel M. Kane, Anthony Ostuni, and Kewen Wu 53:1–53:15

Load Balancing Under Adaptive Bin Deletions
Haim Kaplan, Shay Sapir, and Uri Stemmer 54:1–54:25

Arboricity Matters in Triangle Counting with Random Edges
Arijit Bishnu, Debarshi Chanda, and Gopinath Mishra 55:1–55:25

An Improved Construction of Variety-Evasive Subspace Families
Robert Andrews and Abhibhav Garg 56:1–56:12

Testing Unate Distributions
Daeho Lee, Shivam Nadimpalli, Mingda Qiao, and Ronitt Rubinfeld 57:1–57:24

Homomorphism Testing with Resilience to Online Manipulations
Esty Kelman, Uri Meir, Debanuj Nayak, and Sofya Raskhodnikova 58:1–58:24

On the LSH Distortion of Ulam and Cayley Similarities
Flavio Chierichetti, Mirko Giacchini, Ravi Kumar, and Erasmo Tani 59:1–59:21

Fine-Grained Complexity via Quantum Natural Proofs
Yanlin Chen, Yilei Chen, Rajendra Kumar, Subhasree Patro, and Florian Speelman 60:1–60:23

Locality of Curve-Decoding and Improved Proximity Gaps
Rohan Goyal, Venkatesan Guruswami, Yihang Sun, and Mary Wootters 61:1–61:23

Learning and Testing Convex Functions
Renato Ferreira Pinto Jr., Cassandra Marcussen, Elchanan Mossel, and Shivam Nadimpalli 62:1–62:24

Mixing and Cutoff for the Systematic Scan Dynamics of the Mean-Field Ferromagnetic Potts Model
Antonio Blanca and Md Tahmidur Rafid 63:1–63:22

Lifting Polynomial Complexity Measures Using Error-Correcting Codes
Ivan Hu and Dieter van Melkebeek 64:1–64:10

Property Testing of Computational Networks
Artur Czumaj and Christian Sohler 65:1–65:23

On the Advantage of Adaptivity for Sampling with Cell Probes
Farzan Byramji, Daniel M. Kane, Jackson Morris, and Anthony Ostuni 66:1–66:9

Fast List Recovery of Univariate Multiplicity Codes
Rohan Goyal, Prahladh Harsha, Mrinal Kumar, and Ashutosh Shankar 67:1–67:18

Algorithmic Phase Transition for Large Independent Sets in Dense Hypergraphs
Abhishek Dhawan, Nhi U. Dinh, Eren C. Kızıldağ, Neeladri Maitra, and Bayram A. Şahin 68:1–68:13

QMA Lower Bounds for Batch Verification via Approximate Degree
Mark Bun, Mandar Juvekar, and Samuel King 69:1–69:22

Good Locally Testable Codes with Small Alphabet and Small Query Size
Uriya A. First and Stav Lazarovici 70:1–70:23

Parallel Sampling from the Ising p-Spin Model
Nima Anari, Aniket Das, and Alireza Haqi 71:1–71:23

Quantum Algorithms for Path and Cycle Containment Problems
Arjan Cornelissen, Amin Shiraz Gilani, and Subhasree Patro 72:1–72:23

Testing the Independent Set Property in Hypergraphs
Elena Grigorescu, Shreya Nasa, and Cameron Seth 73:1–73:15

Toward a KKL Theorem for Any HDX
Max Hopkins 74:1–74:14

Improved Local Computation of Edge Orientation
Reut Levi and Bar Rushkin 75:1–75:15

High Probability Streaming Lower Bounds for F_2 Estimation
William Swartworth, David P. Woodruff, and Samson Zhou 76:1–76:23

Approximate Cauchy-Schwarz Inequality and Improved Bounds for Sherali-Adams Refutation of Semirandom CSPs
Pravesh K. Kothari and Andrew D. Lin 77:1–77:20

Local Algorithms and the Failure of Log-Depth Quantum Advantage on Sparse Random CSPs
Antares Chen, Neng Huang, and Kunal Marwaha 78:1–78:26

Preface

This volume contains the papers presented at the 29th International Conference on Approximation Algorithms for Combinatorial Optimization Problems (APPROX 2026) and the 30th International Conference on Randomization and Computation (RANDOM 2026). APPROX focuses on algorithmic and complexity issues surrounding the development of efficient approximate solutions to computationally-difficult problems, and the 2026 edition was the 29th in the series. RANDOM is concerned with applications of randomness to computational and combinatorial problems, and the 2026 edition was the 30th in the series. The two conferences were held in parallel at Boston University, Boston, MA, USA during August 19–21, 2026. This year, the plenary speaker for APPROX was Chandra Chekuri from University of Illinois, Urbaba-Champaign, and the plenary speaker for RANDOM was Madhu Sudan from Harvard University.

Topics of interest for APPROX include approximation algorithms, hardness of approximation, small space, sub-linear time and streaming algorithms, online algorithms, approaches that go beyond worst case analysis, distributed and parallel approximation, embeddings and metric-space methods, mathematical-programming methods, spectral methods, combinatorial optimization, algorithmic game theory, mechanism design and economics, computational geometry problems, approximate learning. Those at RANDOM include the design and analysis of randomized algorithms, randomized complexity theory, pseudorandomness and derandomization, random combinatorial structures, random walks/Markov chains, expander graphs and randomness extractors, probabilistic proof systems, random projections and embeddings, error-correcting codes, average-case analysis, smoothed analysis, property testing, sublinear-time and local algorithms, computational learning theory, and the role of (pseudo)randomness in other areas of computer science such as cryptography, data privacy, and quantum information.

Prior to 2003, APPROX took place in Aalborg (1998), Berkeley (1999), Saarbrücken (2000), Berkeley (2001), and Rome (2002), while RANDOM took place in Bologna (1997), Barcelona (1998), Berkeley (1999), Geneva (2000), Berkeley (2001), and Harvard (2002). Since 2003, APPROX and RANDOM have been co-located, taking place in Princeton (2003), Cambridge (2004), Berkeley (2005), Barcelona (2006), Princeton (2007), Boston (2008), Berkeley (2009), Barcelona (2010), Princeton (2011), Boston (2012), Berkeley (2013), Barcelona (2014), Princeton (2015), Paris (2016), Berkeley (2017), Princeton (2018), Boston (2019), Atlanta (2023), London (2024) and Berkeley (2025). In 2020, 2021, and 2022, the conferences were held online.

The volume contains 30 contributed papers selected by the APPROX Program Committee out of 75 submissions, and 48 contributed papers selected by the RANDOM Program Committee out of 125 submissions. We would like to thank all the authors who submitted papers, the members of the program committees, and the external reviewers. We are grateful for the guidance of the steering committees: Jarosław Byrka, Alina Ene, Samir Khuller, Amit Kumar, Monaldo Mastrolili, Nicole Megow, Laura Sanità, Chaitanya Swamy, László Végh, Virginia Vassilevska Williams, and David P. Williamson for APPROX, and Oded Goldreich, Raghu Meka, Cris Moore, Anup Rao, Omer Reingold, Dana Ron, Noga Ron-Zewi, Ronitt Rubinfeld, Amit Sahai, Ronen Shaltiel, Alistair Sinclair, and Paul Spirakis for RANDOM.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2026).
Editors: Mohit Singh and Tom Gur

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

We would also like to thank the local organizing committee at Boston University Alina Ene, Sofya Raskhodnikova, Adam Smith and the Dagstuhl Publishing Team for their dedicated handling of the proceedings.

It was our pleasure to serve as chairs of the Approx/Random 2026 Program Committees, and to edit this volume of the proceedings.

Mohit Singh and Tom Gur

Program Committees

APPROX 2026

Kristóf Bérczi	Eötvös Loránd University
Karthekeyan Chandrasekaran	University of Illinois, Urbana–Champaign
Leah Epstein	University of Haifa
Arindam Khan	Indian Institute of Science
Euiwoong Lee	University of Michigan
Shi Li	Nanjing University
Konstantin Makarychev	Northwestern University
Ben Moseley	Carnegie Mellon University
Alantha Newman	University of Grenoble Alpes
Neil Olver	London School of Economics
Mohammad Salavatipour	University of Alberta
Guido Schäfer	CWI, Amsterdam
Roy Schwartz	Technion – Israel Institute of Technology
Mohit Singh (PC Chair)	Georgia Institute of Technology
Ali Vakilian	Virginia Tech

RANDOM 2026

Maryam Aliakbarpour	Rice University
Artur Czumaj	University of Warwick
Yotam Dikstein	Institute for Advanced Study
Talya Eden	Bar-Ilan University
Andreas Galanis	University of Oxford
Fernando Granha Jeronimo	University of Illinois Urbana-Champaign
Tom Gur (PC Chair)	University of Cambridge
Nathan Harms	University of British Columbia
Prahladh Harsha	Tata Institute of Fundamental Research
Tali Kaufman	Bar-Ilan University
Oded Lachish	Birkbeck, University of London
Sophie Laplante	IRIF (Université Paris Cité / CNRS)
Chin Ho Lee	North Carolina State University
Amit Levi	University of Haifa
Siqi Liu	Duke University
Or Meir	University of Sheffield
Dor Minzer	Massachusetts Institute of Technology
Sidhanth Mohanty	Northwestern University
Thomas Sauerwald	University of Cambridge
C. Seshadhri	University of California, Santa Cruz
Santhoshini Velusamy	University of Waterloo
Tal Yankovitz	The University of Texas at Austin
Mary Wootters	Stanford University

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2026).
Editors: Mohit Singh and Tom Gur
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Subreviewers

APPROX

Andreas Abels
Marek Adamczyk
Alessandro Aloisio
Stephen Arndt
Yossi Azar
Sayan Bandyapadhyay
Aniket Basu Roy
Sayan Bhattacharya
Ahmad Biniaz
Kimon Boehmer
Prosenjit Bose
Joshua Brakensiek
Silvia Butti
Charlie Carlson
Parinya Chalermsook
Kushagra Chatterjee
Yeshwanth Cherapanamjeri
Marek Chrobak
Jonathan Conroy
Daniel Dadush
Anirban Dasgupta
Bart de Keijzer
Marek Elias
Alina Ene
David Aleman Espinosa
Alireza Farhadi
Yumou Fei
Sándor Fekete
Jeremy Fineman
Zachary Friggstad
Suprovat Ghoshal
Alexander Golovnev
Sathish Govindarajan
Aritanan Gruber
Sariel Har-Peled
Gary Hoppenworth
Richard Huang
Florian Hörsch
Yuni Iwamasa
Mahya Jamshidian
Krishna Kalathur
Danish Kashaev
Yasushi Kawase
Eunjung Kim
Tamás Király
A. Karim Abu-Affash
Sara Ahmadian
Georgios Amanatidis
Per Austrin
Amir Azarmehr
Kiril Bangachev
Aaron Bernstein
Vijay Bhattiprolu
Greg Bodwin
Allan Borodin
Marin Bougeret
Kevin Buchin
Jarek Byrka
Paz Carmi
Hsien-Chih Chang
Xujin Chen
Ashish Chiplunkar
Christian Coester
Martín Costa
Syamantak Das
Mark de Berg
Marina Drygala
Michael Elkin
Thomas Erlebach
Chenglin Fan
Majid Farhadi
Uriel Feige
Moran Feldman
Emily Fox
Mehrdad Ghadiri
Rohan Ghuge
Gramoz Goranci
Fabrizio Grandoni
Waldo Gálvez
Daniel Hathcock
Lingxiao Huang
Yichen Huang
Sharat Ibrahimpur
Rhea Jain
Haotian Jiang
Karthik C. S.
Manas Jyoti Kashyop
Gregory Kehne
Csaba Király
Pieter Kleer

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2026).
Editors: Mohit Singh and Tom Gur
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Sophie Klumper
Pravesh K Kothari
Pooja Kulkarni
Madhumita Kundu
Bundit Laekhanukit
Lap Chi Lau
Asaf Levin
Honghao Lin
Siyue Liu
Chek-Manh Loi
Peter Manohar
Xiao Mao
Frédéric Meunier
Pat Morin
Ramin Mousavi
Meghana Nasre
Yasamin Nazari
Meike Neuwohner
Aidin Niaparast
Taihei Oki
Lorenzo Orecchia
Gyula Pap
Nikos Parotsidis
Kalen Patton
Pan Peng
Michał Pilipczuk
Dömötör Pálvölgyi
Sharath Raghvendra
Rebecca Reiffenhäuser
Rojin Rezvan
Antti Roeyskoe
Heiko Röglin
Samira Samadi
David Saulpic
Kevin Schewior
Jens Schlöter
Tamás Schwarcz
Chris Schwiegelshohn
András Sebő
Supratim Shit
Sudarshan Shyam
Anastasios Sidiropoulos
Stavros Sintos
Balasubramanian Sivan
Shashank Srivastava
Janani Sundaresan
Prafullkumar Tale
Zihan Tan
Yusuke Kobayashi
Ariel Kulik
Nikhil Kumar
An La
Alberto Larrauri
Hung Le
Roie Levin
Alexander Lindermayr
William Lochet
Peter Macgregor
Pasin Manurangsi
Mathieu Mari
Joseph Mitchell
Jonathan Mosheiff
Ns Narayanaswamy
Alfredo Navarra
Ofer Neiman
Heather Newman
Rachit Nimavat
Sebastian Ordyniak
George Osipov
Dmitry Paramonov
Merav Parter
Rameesh Paul
Artur Pessoa
Madhusudhan Pittu
Kent Quanrud
Saladi Rahul
Nicolas Resch
Caleb Robelle
Lars Rohwedder
Harshul Sagar
Sherry Sarkar
Raghuvansh Saxena
Niklas Schlomberg
Melanie Schmidt
Gregory Schwartzman
Michele Scquizzato
Bruce Shepherd
Adi Shraibman
Aaron Sidford
Olha Silina
Rene Sitters
José A. Soto
Noah Stephens-Davidowitz
Daniel Szabo
Karolina Tammemaa
Cuong Than

Stefan Tiegel
Artem Tsikiridis
Rajan Udwani
Kasturi Varadarajan
Karnati Venkata Naga Sreenivasulu
Adrian Vladu
Hoa Vu
Andreas Wiese
Jeff Xu
Yang Yang
Junyao Zhao
Renfei Zhou
Michael Zlatin
Vera Traub
Ta-Wei Tu
Ali Vakilian
Ameya Velingker
Adrian Vetta
Lukas Vogl
Nicole Wein
Chao Xu
Jie Xue
Rico Zenklusen
David Zheng
Weihao Zhu

RANDOM

Tomer Adar
Mohammad Afzali
Yaroslav Alekseev
Vedat Alev
Omar Alrabiah
Konrad Anand
Joel Daniel Andersson
Alexandr Andoni
Robert Andrews
Srinivasan Arunachalam
Dionysis Arvanitakis
Kristina Asimi
Alireza Azizi
Arpon Basu
Sabyasachi Basu
Tugkan Batu
Omri Ben-Eliezer
Ioana-Oriana Bercea
Amey Bhangale
Arnab Bhattacharyya
Aniruddha Biswas
Ari Biswas
Hadley Black
Andrej Bogdanov
Théo Borém Fabris
Joshua Brakensiek
Alex Bredariol Grilo
Chris Brown
Nataly Brukhim
Rares Buhai
Arnav Burudgunte
Sarah Cannon
André Chailloux
Sourav Chakraborty
Siu On Chan
Gautam Chandrasekaran
Ting-Wei Chao
Xiaoyu Chen
Yeyuan Chen
Yu Chen
Kuan Cheng
Ben Cheung
Gil Cohen
Jingqiu Ding
Hang Du
Zeev Dvir
Prateek Dwivedi
David Eppstein
Yumou Fei
Renato Ferreira Pinto Jr.
Yuval Filmus
Eldar Fischer
Daniel Frishberg
William Gay
Sutanu Gayen
Shuyang Gong
Jesse Goodman
Rohan Goyal
Sabee Grewal
Elena Gribelyuk
Zeyu Guo
Mohit Jayanti Gurumukhani
Mursalin Habib
Arran Hamm
Alireza Haqi
Pooya Hatami
Steven Heilman
Max Hopkins
Kaave Hosseini
William Hoza
Jun-Ting Hsieh
Brice Huang
Neng Huang
Enrico Iurlano
Sid Jain
Ce Jin
Satyen Kale
Chandrima Kayal
Zander Kelley
Esty Kelman
Mohammad Khodabandeh
Jung-hun Kim
Shelby Kimmel
Frederic Koehler
Yunbum Kook
Alexander Kulikov
Akash Kumar
Gunjan Kumar
Mrinal Kumar
Vinayak Kumar
Jonathan Leake
Holden Lee
Reut Levi
Baitian Li
Zeyong Li
Jiaming Liang

Jyun-Jie Liao
Hongyang Liu
Bruno Loff
Nikhil Mande
Cassandra Marcussen
Noam Mazor
Mehdi Mhalla
Tushant Mittal
Vishvajeet N
Tejas Nareddy
Justin Oh
Diptaksho Palit
Francisco Pernice
Yash Pote
Amit Rajaraman
Varun Ramanathan
Hanlin Ren
Xuandi Ren
Artur Riazanov
Maurice Rolvien
Adi Rosén
Sai Sandeep
Ramprasad Saptharishi
Raghuvansh Saxena
Tselil Schramm
Sayantan Sen
Nikhil Shagrithaya
Amatya Sharma
Jonathan Shi
Jad Silbak
Makrand Sinha
Dmitry Sokolov
Andrei Staicu
Ria Stevens
John Sylvester
Suguru Tamaki
Kabir Tomer
Alexandra Veliche
Ben Lee Volk
Kangning Wang
Yichuan Wang
Weronika Wrzos-Kaminska
Xinyuan Xie
Helia Yazdanyar
Yuichi Yoshida
Yixiao Yu
Rachel Yun Zhang
Yuming Zhao
Andrew Lin
Yuhan Liu
Shachar Lovett
Gal Maor
Elvira Mayordomo
Davide Mazzali
Gopinath Mishra
Jonathan Mosheiff
Mikito Nanashima
Oanh Nguyen
Hirotaka Ono
Pan Peng
András Pongrácz
Edward Pyne
Prasanna Ramakrishnan
Matt Regehr
Lisheng Ren
Nicolas Resch
Nicolás Rivera
Will Rosenbaum
Sourya Roy
Maya Sankar
Pachara Sawettamalya
Dominik Scheder
Mark Sellke
Cameron Seth
Ronen Shaltiel
Yu-Ching Shen
Abhi Shukul
Noah Singer
Anastasia Sofronova
Shashank Srivastava
Konstantinos Stavropoulos
Janani Sundaresan
Dániel Szabó
Anamay Tengse
Nithin Varma
S. Venkitesh
Erik Waingarten
Robert Wang
Nathan White
Pei Wu
Joy Yang
Junze Yin
Xifan Yu
Hadas Zeilberger
Zihan Zhang
Yan Zhong

List of Authors

Fateme Abbasi (13)
Institute of Computer Science,
University of Wrocław, Poland

Tomer Adar (32, 33)
Technion – Israel Institute of Technology,
Haifa, Israel

Vedat Levi Alev (47)
Department of Mathematics, University of Haifa,
Israel;
Department of Computer Science,
Ariel University, Israel

Hyung-Chan An (13)
Department of Computer Science and
Engineering, Yonsei University, Seoul, South
Korea

Shinwoo An (16)
Bar-Ilan University, Ramat Gan, Israel

Aditya Anand (3)
Computer Science and Engineering,
University of Michigan, Ann Arbor, MI, USA

Nima Anari (71)
Stanford University, CA, USA

Robert Andrews (56)
Cheriton School of Computer Science,
University of Waterloo, Canada

Ilan Doron Arad (5)
Massachusetts Institute of Technology,
Cambridge, MA, USA

Ruben Ascoli (38)
Department of Mathematics, Georgia Institute
of Technology, Atlanta, GA, USA

Amir Azarmehr (43)
Northeastern University, Boston, MA, USA

Ishan Bansal (9)
Amazon, Bellevue, WA, USA

Soheil Behnezhad (43)
Northeastern University, Boston, MA, USA

Huck Bennett (10, 24, 40)
University of Colorado Boulder, CO, USA

Sujoy Bhore (26)
Department of Computer Science & Engineering,
Indian Institute of Technology Bombay, India

Arijit Bishnu (55)
Indian Statistical Institute, Kolkata, India

Antonio Blanca (63)
The Pennsylvania State University,
University Park, PA, USA

Mark Bun (69)
Boston University, MA, USA

Farzan Byramji (37, 66)
UC San Diego, La Jolla, CA, USA

Jarosław Byrka (13)
Institute of Computer Science,
University of Wrocław, Poland

Clément L. Canonne (36, 50)
University of Sydney, Australia

Amir Carmel (30)
Weizmann Institute of Science, Rehovot, Israel

Debarshi Chanda (41, 55)
Indian Statistical Institute, Kolkata, India

Kushagra Chatterjee (22)
Indian Statistical Institute, Kolkata, India

Eshan Chattopadhyay (34)
Cornell University, Ithaca, NY, USA

Shuchi Chawla (17)
Department of Computer Science,
University of Texas at Austin, TX, USA

Chandra Chekuri (12, 15, 21)
University of Illinois, Urbana-Champaign,
IL, USA

Antares Chen (78)
University of Chicago, IL, USA

Ben Chen (39)
School of Computer Science and AI,
Tel-Aviv University, Israel

Yanlin Chen (60)
University of Maryland (QuICS),
College Park, MD, USA

Yilei Chen (60)
Shanghai Qi Zhi Institute,
Tsinghua University, Beijing, China

Joe Cheriyan (9)
Department of Combinatorics and Optimization,
University of Waterloo, Canada

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2026).
Editors: Mohit Singh and Tom Gur
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Flavio Chierichetti (59)
Reddit, San Francisco, CA, USA

Gil Cohen (39, 49)
School of Computer Science and AI,
Tel-Aviv University, Israel

Arjan Cornelissen (72)
Simons Institute, University of California
Berkeley, CA, USA

Artur Czumaj (65)
Department of Computer Science and DIMAP,
University of Warwick, Coventry, UK;
University of Cologne, Germany

Rajni Dabas (8)
Northwestern University, Evanston, IL, USA

Aniket Das (71)
Stanford University, CA, USA

Buddha Dev Das (41)
Indian Statistical Institute, Kolkata, India

Abhishek Dhawan (68)
Department of Mathematics, University of
Illinois Urbana-Champaign, IL, USA

Nhi U. Dinh (68)
Department of Mathematics, University of
Illinois Urbana-Champaign, IL, USA

Dean Doron (39)
Faculty of Computer and Information Science,
Ben-Gurion University of the Negev, Beer Sheva,
Israel

Ilan Doron-Arad (18)
MIT, Cambridge, MA, USA

Yumou Fei (31)
Massachusetts Institute of Technology,
Cambridge, MA, USA

Weiming Feng (51)
School of Computing and Data Science,
The University of Hong Kong, China

Renato Ferreira Pinto Jr. (62)
Columbia University, New York, NY, USA

Arnold Filtser (16, 17)
Bar-Ilan University, Ramat Gan, Israel

Uriya A. First (70)
University of Haifa, Israel

Matthew Fox (24)
University of Colorado Boulder, CO, USA

Bryce Frederickson (38)
Department of Mathematics, Emory University,
Atlanta, GA, USA

Sarah Frederickson (38)
Department of Mathematics, Georgia Institute
of Technology, Atlanta, GA, USA

Yucheng Fu (51)
School of Computing and Data Science,
The University of Hong Kong, China

Theodore Fung (6)
Department of Mathematics, California State
University Northridge, Los Angeles, CA, USA

Karthik Gajulapalli (40)
Georgetown University, Washington, DC, USA

Jonathan Gal (5)
Technion – Israel Institute of Technology,
Haifa, Israel

Junhao Gan (11, 25)
School of Computing and Information Systems,
The University of Melbourne, Australia

Abhibhav Garg (56)
Cheriton School of Computer Science,
University of Waterloo, Canada

Vijay K. Garg (23)
Dept. of Electrical & Computer Engineering,
The University of Texas at Austin, TX, USA

Alma Ghafari (43)
Northeastern University, Boston, MA, USA

Arijit Ghosh (41)
Indian Statistical Institute, Kolkata, India

Suprovat Ghoshal (28)
Indiana University, Bloomington, IN, USA

Mirko Giacchini (59)
Department of Computer Science,
Sapienza University of Rome, Italy

Amin Shiraz Gilani (72)
QuICS, University of Maryland,
College Park, MD, USA

Alexander Golovnev (40)
Georgetown University, Washington, DC, USA

Rohan Goyal (61, 67)
Massachusetts Institute of Technology,
Cambridge, MA, USA

Svenja M. Griesbach (27)
Department of Computer Science,
RWTH Aachen University, Germany

Elena Grigorescu (73)
Cheriton School of Computer Science,
University of Waterloo, Canada

Mara Grilnberger (29)
Department of Computer Science,
University of Salzburg, Austria

Mohit Gurumukhani (34)
Cornell University, Ithaca, NY, USA

Venkatesan Guruswami (19, 20, 61)
Simons Institute for the Theory of Computing,
University of California, Berkeley, CA, USA;
Departments of EECS and Mathematics,
University of California, Berkeley, CA, USA

Yi Han (42)
Institute for Advanced Study,
Princeton, NJ, USA

Alireza Haqi (71)
Stanford University, CA, USA

Themistoklis Haris (52)
Department of Computer Science,
Boston University, MA, USA

Prahladh Harsha (67)
Tata Institute of Fundamental Research,
Mumbai, India

Shuichi Hirahara (46)
National Institute of Informatics, Tokyo, Japan

Felix Hommelsheim (7, 27)
University of Cologne, Germany

Max Hopkins (74)
Institute for Advanced Study,
Princeton, NJ, USA

Yahel Hotam (33)
University of Haifa, Israel

Ivan Hu (64)
University of Wisconsin-Madison, WI, USA

Neng Huang (28, 78)
University of Michigan, Ann Arbor, MI, USA

Rhea Jain (15)
Siebel School of Computing and Data Science,
University of Illinois, Urbana-Champaign, IL,
USA

Mandar Juvekar (69)
Boston University, MA, USA

Daniel M. Kane (37, 53, 66)
UC San Diego, La Jolla, CA, USA

Haim Kaplan (54)
Tel Aviv University, Israel;
Google Research, Tel Aviv, Israel

Esty Kelman (58)
Massachusetts Institute of Technology,
Cambridge, MA, USA;
Boston University, MA, USA

Yuval Khaskelberg (39)
School of Computer Science and AI,
Tel-Aviv University, Israel

Samir Khuller (8)
Northwestern University, Evanston, IL, USA

Samuel King (69)
Georgetown University, Washington, DC, USA

Max Klimm (27)
Institute of Mathematics, Technische Universität
Berlin, Germany

Pravesh K. Kothari (77)
Department of Computer Science,
Princeton University, NJ, USA

Robert Krauthgamer (30)
The Harry Weinrebe Professorial Chair of
Computer Science, Weizmann Institute of
Science, Rehovot, Israel

Pooja Kulkarni (21)
University of Chicago, IL, USA

Mrinal Kumar (67)
Tata Institute of Fundamental Research,
Mumbai, India

Rajendra Kumar (60)
Indian Institute of Technology Delhi,
New Delhi, India

Ravi Kumar (2, 59)
Google Research, Mountain View, CA, USA

Nick Kushnir (35)
School of Mathematics,
Tel Aviv University, Israel

Eren C. Kızıldağ (68)
Department of Statistics, University of Illinois
Urbana-Champaign, IL, USA

Stav Lazarovici (70)
University of Haifa, Israel

Changyeol Lee (13)
Department of Computer Science and
Engineering, Yonsei University, Seoul, South
Korea

Daeho Lee (57)
Massachusetts Institute of Technology,
Cambridge, MA, USA

Euiwoong Lee (28)
University of Michigan, Ann Arbor, MI, USA

Amit Levi (33)
University of Haifa, Israel

Reut Levi (75)
Reichman University, Herzliya, Israel

Roie Levin (2)
Rutgers University, Piscataway, NJ, USA

Andrew D. Lin (77)
Program in Applied and Computational Mathematics, Princeton University, NJ, USA

Bingkai Lin (20)
State Key Laboratory of Novel Software Technology, Nanjing University, China

Nati Linial (48)
School of Computer Science and Engineering, Hebrew University, Jerusalem, Israel

Elyassaf Loyfer (48)
School of Computer Science and Engineering, Hebrew University, Jerusalem, Israel

Peter Ly (10)
University of Colorado Boulder, CO, USA

Neeladri Maitra (68)
Department of Mathematics, University of Illinois Urbana-Champaign, IL, USA

Konstantin Makarychev (28)
Northwestern University, Evanston, IL, USA

Yury Makarychev (3, 28)
Toyota Technological Institute at Chicago (TTIC), IL, USA

Pasin Manurangsi (1)
Google Research, Bangkok, Thailand

Gal Maor (49)
Tel Aviv University, Israel

Cassandra Marcussen (62)
Harvard University, Cambridge, MA, USA

Kunal Marwaha (78)
University of Chicago, IL, USA

Caleb McFarland (38)
Department of Mathematics, Georgia Institute of Technology, Atlanta, GA, USA

Ruta Mehta (21)
University of Illinois at Urbana Champaign, IL, USA

Uri Meir (58)
Blavatnik School of Computer Science, Tel Aviv University, Israel

Gopinath Mishra (41, 55)
Institute of Mathematical Sciences, Chennai, India

Bryant Morrell (24)
University of Colorado Boulder, CO, USA

Jackson Morris (37, 66)
UC San Diego, La Jolla, CA, USA

Elchanan Mossel (62)
MIT, Cambridge, MA, USA

Shivam Nadimpalli (57, 62)
Massachusetts Institute of Technology, Cambridge, MA, USA

Satyadev Nandakumar (44)
Department of Computer Science and Engineering, Indian Institute of Technology Kanpur, India

Joseph (Seffi) Naor (2, 5)
Technion – Israel Institute of Technology, Haifa, Israel

Shreya Nasa (73)
Department of Computer Science, Purdue University, West Lafayette, IN, USA

Guyslain Naves (12)
Aix-Marseille Univ, CNRS, LIS, Marseille, France

Debanuj Nayak (58)
Boston University, MA, USA

Nimitt (36)
IIT Gandhinagar, India

Kazuki Ogitsuka (46)
The Graduate University for Advanced Studies (SOKENDAI), Tokyo, Japan

Anthony Ostuni (37, 53, 66)
UC San Diego, La Jolla, CA, USA

Diptaksho Palit (52)
Department of Computer Science, Boston University, MA, USA

Debmalya Panigrahi (2)
Duke University, Durham, NC, USA

Ori Parzanchevski (47)
Einstein Institute of Mathematics, Hebrew University of Jerusalem, Israel

Subhasree Patro (60, 72)
Eindhoven University of Technology,
The Netherlands

Michał Pawłowski (26)
Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Poland;
Department of Computer, Control and Management Engineering, Sapienza University of Rome, Italy;
IDEAS NCBR, Warsaw, Poland

Nir Petruschka (30)
Weizmann Institute of Science, Rehovot, Israel

Joseph Poremba (12)
Department of Computer Science, University of British Columbia, Vancouver, Canada

Logan Post (38)
Department of Mathematics, Georgia Institute of Technology, Atlanta, GA, USA

Yash Pote (50)
National University of Singapore, Singapore

Subin Pulari (44)
National Research University Higher School of Economics Moscow, Russia

Mingda Qiao (57)
University of Massachusetts Amherst, MA, USA

Md Tahmidur Rafid (63)
The Pennsylvania State University,
University Park, PA, USA

Rajiv Raman (14)
Indraprastha Institute of Information Technology Delhi, New Delhi, India

Sofya Raskhodnikova (58)
Boston University, MA, USA

Xuandi Ren (19, 20)
Department of EECS, University of California, Berkeley, CA, USA

Rojin Rezvan (22)
Virginia Tech, Blacksburg, VA, USA

Noam Ringach (34)
Cornell University, Ithaca, NY, USA

Emilie Rivkin (8)
Northwestern University, Evanston, IL, USA

Ronitt Rubinfeld (57)
Massachusetts Institute of Technology,
Cambridge, MA, USA

Bar Rushkin (75)
Reichman University, Herzliya, Israel

Akhil S (44)
Department of Computer Science and Engineering, Indian Institute of Technology Kanpur, India

Bayram A. Şahin (68)
Department of Statistics, University of Illinois Urbana-Champaign, IL, USA

Shay Sapir (54)
Weizmann Institute, Rehovot, Israel;
Google Research, Tel Aviv, Israel

Siddhartha Sarkar (14)
Indian Institute of Science, Bengaluru, India

Suronjona Sarma (44)
Department of Computer Science and Engineering, Indian Institute of Technology Kanpur, India

Jonathan Scarlett (50)
National University of Singapore, Singapore

Cameron Seth (73)
Cheriton School of Computer Science,
University of Waterloo, Canada

Hadas Shachnai (18)
Technion – Israel Institute of Technology,
Haifa, Israel

Liren Shan (3)
Toyota Technological Institute at Chicago (TTIC), IL, USA

Ashutosh Shankar (67)
Tata Institute of Fundamental Research,
Mumbai, India

Asaf Shapira (35)
School of Mathematics,
Tel Aviv University, Israel

F. Bruce Shepherd (12)
Department of Computer Science, University of British Columbia, Vancouver, Canada

Kristin Sheridan (17)
Department of Computer Science,
University of Texas at Austin, TX, USA

Yongho Shin (13)
Institute of Computer Science,
University of Wrocław, Poland

Gilad Shmerler (18)
Technion – Israel Institute of Technology,
Haifa, Israel

Miles Simmons (9)
Department of Combinatorics and Optimization, University of Waterloo, Canada

Antonis Skarlatos (29)
Department of Computer Science, University of Warwick, Coventry, UK

Christian Sohler (65)
University of Cologne, Germany

Florian Speelman (60)
University of Amsterdam and QuSoft, The Netherlands

Uri Stemmer (54)
Tel Aviv University, Israel; Google Research, Tel Aviv, Israel

Robert P. Streit (23)
Dept. of Electrical & Computer Engineering, The University of Texas at Austin, TX, USA

Madhu Sudan (43)
Harvard University, Cambridge, MA, USA

Xiao Sun (11, 25)
School of Computing and Information Systems, The University of Melbourne, Australia

Yihang Sun (61)
Stanford University, CA, USA

William Swartworth (76)
Carnegie Mellon University, Pittsburgh, PA, USA

Amnon Ta-Shma (39)
School of Computer Science and AI, Tel-Aviv University, Israel

Shaoxuan Tang (19)
Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China

Erasmo Tani (59)
Department of Computer Science, Sapienza University of Rome, Italy

Yonatan Trachtenberg (17)
Department of Computer Science, Bar-Ilan University, Ramat Gan, Israel

Csaba D. Tóth (6)
Department of Mathematics, California State University Northridge, Los Angeles, CA, USA; Department of Computer Science, Tufts University, Medford, MA, USA

Seeun William Umboh (11, 25, 26)
School of Computing and Information Systems, The University of Melbourne, Australia; ARC Training Centre in Optimisation Technologies, Integrated Methodologies, and Applications (OPTIMA), Melbourne, Australia

Ali Vakilian (22)
Virginia Tech, Blacksburg, VA, USA

Dieter van Melkebeek (64)
University of Wisconsin-Madison, WI, USA

Santhoshini Velusamy (4)
David R. Cheriton School of Computer Science, University of Waterloo, Canada

Jan Vondràk (21)
Stanford University, CA, USA

Evelyn Warton (40)
Oregon State University, Corvallis, OR, USA

David P. Woodruff (76)
Carnegie Mellon University, Pittsbrugh, PA, USA

Mary Wootters (61)
Stanford University, CA, USA

Weronika Wrzos-Kaminska (45)
EPFL, Lausanne, Switzerland

Kewen Wu (53)
Caltech, Pasadena, CA, USA

Jatin Yadav (14)
Indian Institute of Technology Delhi, New Delhi, India

Joy Qiping Yang (50)
University of Sydney, Australia

Minji Yang (51)
School of Computing and Data Science, The University of Hong Kong, China

Anqi Zhang (51)
The Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China

Yunya Zhao (34)
Cornell University, Ithaca, NY, USA

Xin Zheng (20)
State Key Laboratory of Novel Software Technology, Nanjing University, China

Samson Zhou (76)
Texas A&M University, College Station, TX, USA

Improved Approximation Algorithm for Maximum Balanced Biclique

Pasin Manurangsi
Google Research, Bangkok, Thailand

Abstract

We study the Maximum Balanced Biclique (MBB) problem: Given a bipartite graph G with n vertices on each side, find a balanced biclique in G with maximum size. We give a polynomial-time $\left(\frac{n}{\widetilde{\Omega}\left((\log n)^3\right)}\right)$-approximation algorithm for the problem, which improves upon an $\left(\frac{n}{\Omega\left((\log n)^2\right)}\right)$-approximation by Chalermsook et al. [8] and answers their open question. Furthermore, our approximation ratio matches that of the maximum clique problem by Feige [11] up to an $O(\log \log n)$ factor.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis; Theory of computation → Graph algorithms analysis; Theory of computation → Discrete optimization

Keywords and phrases Maximum Balanced Biclique, Approximation Algorithm, Semidefinite Programming

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.1

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2604.27141

1 Introduction

The *Maximum Clique* problem – where we are given an undirected graph and the goal is to find a maximum clique – is one of the most well studied problems in all of combinatorial optimization. One of the original 21 NP-hard problems shown by Karp [20], Maximum Clique also played a central role in the foundation for connections between the PCP Theorem and hardness of approximation [12, 2, 3, 5, 18, 21, 9, 23, 29]. Specifically, it is NP-hard to approximate Maximum Clique to within a factor of $n^{1-\epsilon}$ for any constant $\epsilon > 0$ [18, 29]. Under a stronger assumption that NP $\not\subseteq$ BPTIME$\left(2^{(\log n)^{O(1)}}\right)$, Khot and Ponnuswami [23] ruled out any polynomial-time algorithm with approximation ratio $n/2^{(\log n)^{3/4+\gamma}}$ for any constant $\gamma > 0$. This remains the best known hardness of approximation for Maximum Clique to date. Meanwhile, many approximation algorithms for Maximum Clique have been devised (e.g. [7, 1, 11]). In particular, Feige's algorithm [11] achieves the best known approximation ratio of $O\left(n \cdot \frac{(\log \log n)^2}{(\log n)^3}\right)$ in polynomial time.

In this work, we study a bipartite variant of the Maximum Clique problem, called the *Maximum Balanced Biclique (MBB)* problem. Recall that a k-biclique, denoted by $K_{k,k}$, is the complete bipartite subgraph where each side contains k vertices. In MBB, we are given a bipartite graph $G = (U, V, E)$ where $|U| = |V| = n$, and the goal is to find a maximum k-biclique in G.

The exact version of MBB was stated[1] to be NP-hard in Garey and Johnson's seminal book [16, page 196]. However, despite a considerable amount of effort over the years (e.g. [10, 13, 22, 26, 6, 27]), it is *not* even known to be NP-hard to approximate to within

[1] Originally, the book did not provide a proof, but such a proof was later published e.g. in [19].

© Pasin Manurangsi;
licensed under Creative Commons License CC-BY 4.0

a factor of $1+\epsilon$ for any constant $\epsilon > 0$. Nevertheless, hardness of approximation results are known under other assumptions [10, 13, 22, 26, 6, 27]. Specifically, it is known to be NP-hard (under randomized reductions) to approximate to within a factor of $n^{1-\epsilon}$ for any $\epsilon > 0$, assuming strengthenings of the Unique Games Conjecture [6, 27].

Despite the apparent challenges in proving hardness of approximation results for MBB, not much is known in terms of approximation algorithms either. It is straightforward to see that an "enumeration style" algorithm yields $\left(\frac{n}{\tilde{\Omega}(\log n)}\right)$-approximation for MBB in polynomial time. Only recently did Chalermsook et al. [8] manage to improve upon this approximation guarantee, obtaining an $\left(\frac{n}{\tilde{\Omega}((\log n)^2)}\right)$-approximation. As an open question, they asked whether it is possible to improve this guarantee to match the $\left(\frac{n}{\tilde{\Omega}((\log n)^3)}\right)$-approximation by [11] for the Maximum Clique problem.

1.1 Our Contribution

Our main result is an algorithm that achieves such an approximation guarantee:

▶ **Theorem 1** (Main Theorem). *There is a randomized polynomial-time $O\left(n \cdot \left(\frac{\log\log n}{\log n}\right)^3\right)$-approximation for Maximum Balanced Biclique.*

Our approximation ratio matches the aforementioned algorithm of Feige for the Maximum Clique [11] problem up to an $O(\log\log n)$ factor.

Proof Overview

To explain the proof techniques, let us use k to denote the size of the optimal balanced biclique in G, and let $t = n/k$. We first review the algorithmic guarantee of [8]: If $t \leq (\log n)^{O(1)}$, then they show how to use graph removal technique from [11] on *matchings* in order to find $\tilde{\Omega}((\log n)^2)$-biclique, as formalized below.

▶ **Theorem 2** ([8]). *Given a graph G that contains a k-biclique, there is a polynomial-time algorithm that finds a $\Theta\left(\left(\frac{\log n}{\log t}\right)^2\right)$-biclique in G where $t = n/k$.*

It is not hard to see that the above theorem immediately yields $O(n/(\log n)^2)$-approximation. For our purpose, observe further that, if $t \geq \tilde{\Omega}(\log n)$, then the above already yields $\left(\frac{n}{\tilde{\Omega}((\log n)^3)}\right)$-approximation. Thus, we can focus on the case $t \leq \tilde{O}(\log n)$, i.e. when the optimum k is at least $\frac{n}{\tilde{O}(\log n)}$.

For the case of Maximum Clique, this case can be solved via semi-definite program (SDP) rounding [1, 17] or via subgraph removal [7]. In particular, for $t \leq O\left(\frac{\log n}{\log\log n}\right)$, these algorithms can find a clique of size $n^{1/O(t)}$. Unfortunately, Chalermsook et al.'s [8] technique of reducing MBB to Maximum Clique has a square blow up on the parameter t. This means that one can only apply these Maximum Clique algorithms in a black-box manner[2] when $t \leq \tilde{O}(\sqrt{\log n})$, which is not sufficient for our purpose.

Instead, our main result is to show that, in the regime $t \leq O\left(\frac{\log n}{\log\log n}\right)$, a more direct SDP relaxation and rounding can yield a balanced biclique of size $n^{1/O(t)}$, as stated below. As mentioned earlier, combining this with Chalermsook et al.'s result (Theorem 2) immediately yields the final approximation guarantee.

[2] See [28] for a more detailed description on how to interpret [8] as a reduction from MBB to Maximum Clique.

▶ **Theorem 3** (Main SDP Rounding). *There exists a constant $D > 1$ such that the following holds. If $\{u_i\}, e$ is a feasible solution to the SDP relaxation in Figure 2 with $k \geq n \cdot \frac{D \cdot \log\log n}{\log n}$, there is an efficient rounding algorithm for finding a balanced biclique of at least size $n^{\frac{1}{D \cdot t}}$ with high probability, where $t = n/k$.*

Integrality Gap for A Natural Relaxation

To illustrate a challenge faced when using SDP relaxation for MBB, let us first state the following "natural" extension of Lovasz's ϑ function [25], which is used[3] for Maximum Clique [1, 17]. Here, we associate a vector $u_i \in \mathbb{R}^d$ with each $i \in U \cup V$, and let $e \in \mathbb{R}^d$ be an arbitrary unit vector.

$$
\begin{aligned}
\text{Maximize} \quad & k \\
\text{Subject to} \quad & \|e\|^2 = 1 \\
& \|u_i\|^2 = \langle u_i, e \rangle && \forall i \in U \cup V \\
& \sum_{i \in U} \langle u_i, e \rangle = \sum_{j \in V} \langle u_j, e \rangle = k && (1) \\
& \langle u_i, u_j \rangle = 0 && \forall (i,j) \in (U \times V) \setminus E \\
& \langle u_i, u_j \rangle \geq 0 && \forall (i,j) \in U \times V
\end{aligned}
$$

Figure 1 SDP Relaxation for MBB (First Attempt).

If G contains $K_{k,k}$, there is a feasible solution (setting vectors in the biclique to e and all others to 0) of value k. We remark that the only difference compared to the Maximum Clique formulation here is the constraint (1), which aims to ensure that the number of vertices picked in U is the same as that in V (so that the biclique is balanced).

Unfortunately, this SDP relaxation is completely unhelpful. We can always find a solution with $k = n/2$: Let f be any unit vector orthogonal to e, and let $u_i = \frac{e+f}{2}$ for all $i \in U$ and $u_j = \frac{e-f}{2}$ for all $j \in V$. This ensures that $\langle u_i, u_j \rangle = 0$ for all $(i,j) \in U \times V$. In other words, the integrality gap is unbounded.

Our SDP Relaxation

Given the integrality gap, we thus use a stronger SDP relaxation for MBB. We again associate a vector $u_i \in \mathbb{R}^d$ with each $i \in U \cup V$, and let $e \in \mathbb{R}^d$ be a unit vector. The SDP relaxation is stated below. Note that we formulate it as a feasibility problem rather than an optimization problem.

We note that the only additional constraints are (5) and (6), which are "fractional degree" constraints aiming to ensure that if i is picked, then k of its neighbor must be picked. Again, if G contains $K_{k,k}$, there is a feasible solution (setting vectors in the biclique to e and all others to 0).

[3] Note that the formulation below looks different from one defined in [25] but the latter can be written in this form [24].

$$
\begin{aligned}
&\|e\|^2 = 1 && && \\
&\|u_i\|^2 = \langle u_i, e\rangle && \forall i \in U \cup V && \\
&\sum_{i\in U}\langle u_i, e\rangle = \sum_{j\in V}\langle u_j, e\rangle = k && && (2)\\
&\langle u_i, u_j\rangle = 0 && \forall (i,j) \in (U \times V) \setminus E && (3)\\
&\langle u_i, u_j\rangle \geq 0 && \forall (i,j) \in U \times V && (4)\\
&\sum_{j\in V}\langle u_i, u_j\rangle = k\langle u_i, e\rangle && \forall i \in U && (5)\\
&\sum_{i\in U}\langle u_i, u_j\rangle = k\langle u_j, e\rangle && \forall j \in V && (6)
\end{aligned}
$$

Figure 2 SDP Relaxation for MBB.

Our main technical contribution (Theorem 3) is to show that a feasible solution to this SDP can be rounded to a large biclique. It turns out that the rounding procedure is essentially the same as that of [17] in the clique case: After some preprocessing, pick a random Gaussian g and keep a vertex i if its dot product with g is at least a certain threshold. (The full procedure is presented in Figure 3.)

The main difference however is in the analysis. In [17], one simply computes the expected number of vertices that survive the rounding and the expected survived number of non-edge pairs. If the former is much larger than the latter, then (with not too small probability) we can find a large clique by deleting one end of every non-edge pair. This analysis does not work for us: Even if the expected number of survived vertices is large, it is possible that the rounding either keep all vertices in U and no vertex in V, or vice versa. Indeed, this is the situation that occurs with the above integrality gap instance of the SDP in Figure 1.

To overcome this issue, we instead lower bound the expected number of survived *edges*. We do this by showing that, due to the extra constraints (5) and (6), many edges $(i,j) \in E$ are "good" in the sense that $\langle u_i, u_j\rangle$ is not too small. This implies that they are positively correlated in the rounding procedure, which allows us to lower bound the expected number of survived edges. This conclude our proof overview for the rounding procedure, which is formalized in Section 3 after some preliminaries.

2 Preliminaries

Throughout this work, we use log to denote the natural logarithm.

2.1 Gaussian Tail Bounds

Our rounding scheme relies on probability bounds for (multivariate) normal distributions, which we will state here for convenience. We note that these bounds are standard and are also used in approximation algorithms for Maximum Clique (e.g. [17]). First, let us start with the following lemma:

▶ **Lemma 4** ([15]). *Let $\phi(x) = \frac{1}{\sqrt{2\pi}} \cdot \exp(-x^2/2)$ be the PDF of the standard Gaussian distribution $\mathcal{N}(0,1)$. If $Z \sim \mathcal{N}(0,1)$ and $\tau \geq 2$, then $\frac{\phi(\tau)}{2\tau} \leq \Pr[Z \geq \tau] \leq \frac{\phi(\tau)}{\tau}$.*

For two standard Gaussian variables X, Y with correlation $\rho = \mathbb{E}[XY]$, we use the following bivariate tail bounds for $\Pr[X \geq \tau, Y \geq \tau]$ for $\tau \geq 2$:

- **Positive Correlation ($\rho \geq 0$):** Since X, Y are positively correlated, the joint probability is lower-bounded by the product of their marginals:

$$\Pr[X \geq \tau, Y \geq \tau] \geq \Pr[X \geq \tau] \Pr[Y \geq \tau] \geq \frac{\phi(\tau)^2}{4\tau^2}, \tag{7}$$

  where the last inequality is due to Lemma 4.
- **Negative Correlation ($\rho < 0$):** Notice that $\frac{X+Y}{\sqrt{2+2\rho}} \sim \mathcal{N}(0,1)$. Thus,

$$\Pr[X \geq \tau, Y \geq \tau] \leq \Pr[X + Y \geq 2\tau] < \frac{1}{\sqrt{2}\tau} \cdot \phi\left(\frac{\sqrt{2} \cdot \tau}{\sqrt{1+\rho}}\right) \leq \frac{\sqrt{\pi}}{\tau} \cdot \phi(\tau)^2 \cdot \exp\left(\rho \cdot \tau^2\right) \tag{8}$$

  where the first inequality is due to Lemma 4 and $\rho < 0$, and the second inequality follows from $\rho < 0$.

2.2 Biclique Extraction Lemma

Before we describe the rounding, let us also state a self-contained combinatorial step of extracting a balanced biclique from a sufficiently dense surviving subgraph.

▶ **Lemma 5** (Greedy Extraction). *Let $H = (U_H, V_H, E_H)$ be a bipartite graph with at most n vertices on each side. Let F be the total number of edges (i.e. $F = |E_H|$) and Q be the total number of non-edges in H (i.e. $Q = |U_H| \cdot |V_H| - |E_H|$). If there exists an integer $r \geq 1$ such that $F - 2rQ \geq 2nr$, then there is a polynomial-time algorithm that finds a $K_{r,r}$ in H.*

Proof. Let $d_F(i)$ and $d_Q(i)$ be the degrees of a vertex i in the edges and non-edges of H, respectively. We apply the following two-stage procedure:

1. **Density Cleaning:** While there exists any vertex i such that $d_F(i) \leq 2r \cdot d_Q(i)$, delete i from H.

   To analyze this procedure, we track $W = F - 2rQ$. Upon deleting i, the change in W is $-d_F(i) + 2r \cdot d_Q(i) \geq 0$. Thus, the final cleaned graph $H' = (U_{H'}, V_{H'}, E_{H'})$ satisfies $W' \geq 2nr$.

   Since $W' \leq |E_{H'}| \leq |U_{H'}| \cdot |V_{H'}|$ and we assume that each side of H is of size at most n, we can conclude that $|U_{H'}|, |V_{H'}|$ are both at least $2r$.
2. **Biclique Construction:** We arbitrarily select exactly r vertices from $U_{H'}$ to form U_{cliq}. For each selected $u \in U_{cliq}$, we delete its non-neighbors from $V_{H'}$; from how H' is constructed, there are at most $\frac{|V_{H'}|}{2r}$ such non-neighbors. Thus, the total number of vertices deleted from $V_{H'}$ is at most $r\left(\frac{|V_{H'}|}{2r}\right) = \frac{|V_{H'}|}{2}$. Consequently, the number of vertices surviving in $V_{H'}$ is at least $\frac{|V_{H'}|}{2}$. Because we established $|V_{H'}| \geq 2r$, at least $\frac{2r}{2} = r$ vertices remain. We arbitrarily select r of these surviving vertices to form V_{cliq}. The set $U_{cliq} \cup V_{cliq}$ is the desired balanced biclique. ◀

We remark that, for the regime $\Theta(1) \leq r \leq n^{1-\Omega(1)}$, known techniques [14, 4] can be used in the "Biclique Construction" step to improve the bound from r to $\Omega(r \log n)$. However, this does not affect the overall approximation ratio of our algorithm and, thus, we opt for the argument above for simplicity.

3 The Rounding Algorithm and Proof of Theorem 3

The full SDP rounding procedure is presented in Figure 3.

1. **SDP:** Let $\{u_i\}, e$ be vectors satisfying Figure 2. Let $t = n/k$, and $c_i = \langle u_i, e\rangle$ for all $i \in U \cup V$.
2. **Heavy Sets:** Let $A = \{i \in U \mid c_i \geq \frac{1}{8t}\}$ and $B = \{j \in V \mid c_j \geq \frac{1}{8t}\}$.
3. **Shifting:** Let $\alpha = 1 - 1/\sqrt{2}$, and let $u'_i = u_i - \alpha c_i e$ for all $i \in A \cup B$.
4. **Gaussian Thresholding:** Sample $g \sim \mathcal{N}(0, I_d)$. Keep vertices $i \in A \cup B$ iff $X_i := \langle g, \frac{u'_i}{\|u'_i\|}\rangle \geq \tau$, where $\tau = \sqrt{0.1 \log n}$. Let the surviving induced subgraph be H.
5. **Biclique Extraction:** Run the procedure from Lemma 5 on H, and output the result.

Figure 3 SDP Rounding Algorithm for MBB.

Proof of Theorem 3. We will next analyze the guarantee of the rounding algorithm. To do so, let $D = 1000$ and assume that $t \leq \frac{\log n}{D \cdot \log\log n}$.

For any subsets $S \subseteq U$ and $T \subseteq V$, let $M(S,T) := \sum_{(i,j)\in S\times T}\langle u_i, u_j\rangle$. We have

$$\begin{aligned}
M(A,B) &= M(U,V) - M(U\setminus A, V) - M(U, V\setminus B) + M(U\setminus A, V\setminus B)\\
&\overset{(4)}{\geq} M(U,V) - M(U\setminus A, V) - M(U, V\setminus B)\\
&= \sum_{i\in U}\left(\sum_{j\in V}\langle u_i,u_j\rangle\right) - \sum_{i\in U\setminus A}\left(\sum_{j\in V}\langle u_i,u_j\rangle\right) - \sum_{j\in V\setminus B}\left(\sum_{i\in U}\langle u_i,u_j\rangle\right)\\
&\overset{(5),(6)}{=} \sum_{i\in U} k\cdot c_i - \sum_{i\in U\setminus A} k\cdot c_i - \sum_{j\in V\setminus B} k\cdot c_j\\
&\overset{(2)}{=} k^2 - \sum_{i\in U\setminus A} k\cdot c_i - \sum_{j\in V\setminus B} k\cdot c_j\\
&\overset{(\star)}{\geq} k^2 - n\cdot k\cdot\frac{1}{8t} - n\cdot k\cdot\frac{1}{8t} \qquad = \frac{3}{4}k^2,
\end{aligned} \tag{9}$$

where $(\star)$ follows from our definition of A, B.

Let $E^+ = \{(i,j) \in A\times B \mid \langle u_i, u_j\rangle > \frac{1}{2}c_i c_j\}$.

Note that $\sum_{(i,j)\in A\times B} c_i c_j \leq (\sum_U c_i)(\sum_V c_j) \overset{(2)}{=} k^2$. Hence,

$$\sum_{(i,j)\in A\times B}\left(\langle u_i,u_j\rangle - \frac{1}{2}c_i c_j\right) \geq M(A,B) - \frac{k^2}{2} \overset{(9)}{\geq} \frac{k^2}{4}$$

Any pair $(i,j) \notin E^+$ contributes a non-positive value to this sum. This, together with the fact that $\langle u_i, u_j\rangle \leq 1$ for all $i, j \in U \cup V$, implies that $|E^+| \geq \frac{k^2}{4}$. Note also that, by (3), all pairs $(i,j) \in E^+$ must be an edge.

Next, consider any $i, j \in U \cup V$. We have $\langle u'_i, u'_j\rangle = \langle u_i, u_j\rangle - \frac{1}{2}c_i c_j$. Note that this also implies $\|u'_i\|_2^2 = c_i(1 - \frac{1}{2}c_i) \leq c_i$. We next analyze the expected number of edges and non-edges in H. To do this, let F denote the number of surviving edges in H, Q denote the number of non-edges in H, and let

$$r = \frac{1}{64\sqrt{\pi}\cdot\tau\cdot t^2}\cdot\exp\left(\frac{\tau^2}{16t}\right).$$

Note that, for sufficiently large n, we have

$$r \geq \frac{1}{O((\log n)^3)} \cdot n^{\frac{1}{160t}} \geq n^{\frac{1}{D \cdot t}}. \tag{10}$$

- **Lower Bounding $\mathbb{E}[F]$:** For $(i,j) \in E^+$, since $\langle u_i, u_j \rangle \geq \frac{1}{2} c_i c_j$, we have $\langle u'_i, u'_j \rangle \geq 0$. Thus, $\mathbb{E}[X_i X_j] \geq 0$. Applying (7), the probability that (i,j) is kept in H is at least $\frac{\phi(\tau)^2}{4\tau^2}$. Hence,

$$\mathbb{E}[F] \geq |E^+| \cdot \frac{\phi(\tau)^2}{4\tau^2} \geq \frac{k^2}{4} \cdot \frac{\phi(\tau)^2}{4\tau^2}. \tag{11}$$

  Since $k = n/t$ and due to our bound on t, for any sufficiently large n, we have

$$\mathbb{E}[F] \geq \frac{n^2}{O((\log n)^3)} \cdot \exp(-0.1 \log n) \geq n^{1.8} \geq n \cdot \exp\left(\frac{\tau^2}{16t}\right) \geq 8nr. \tag{12}$$

- **Upper Bounding $\mathbb{E}[Q]$:** Consider any $(i,j) \in (A \times B) \setminus E$. By (3), $\langle u_i, u_j \rangle = 0$, which implies $\langle u'_i, u'_j \rangle = -\frac{1}{2} c_i c_j$. Normalizing bounds the correlation:

$$\mathbb{E}[X_i X_j] = \left\langle \frac{u'_i}{\|u'_i\|}, \frac{u'_j}{\|u'_j\|} \right\rangle \leq \frac{-\frac{1}{2} c_i c_j}{\sqrt{c_i c_j}} = -\frac{1}{2}\sqrt{c_i c_j} \leq -\frac{1}{16t},$$

  where the last inequality follows from definition of A, B. Applying (8), we have

$$\mathbb{E}[Q] \leq n^2 \cdot \frac{\sqrt{\pi}}{\tau} \cdot \phi(\tau)^2 \cdot \exp\left(-\frac{\tau^2}{16t}\right) \overset{(11)}{=} \frac{1}{4r} \cdot \mathbb{E}[F]. \tag{13}$$

Thus, we have

$$\mathbb{E}[F - 2rQ] \overset{(13)}{\geq} \mathbb{E}[F]/2 \overset{(12)}{\geq} 4nr.$$

Since $F - 2rQ$ has value at most n^2, we have $F - 2rQ \geq 2nr$ with probability at least $1/n^2$. When this occurs, Lemma 5 guarantees an r-biclique which satisfies the desired size bound due to (10). To achieve high probability, we simply repeat the rounding process n^3 times. This concludes our proof. ◀

4 Putting Things Together: Proof of Theorem 1

Finally, we prove our main theorem (Theorem 1) by combining our SDP rounding (in the large optimum case) with the algorithm of Chalermsook et al. from Theorem 2 (in the small optimum case).

Proof of Theorem 1. We solve the SDP from Figure 2 for $k = \left\lceil n \cdot \frac{10 \cdot D \cdot \log\log n}{\log n} \right\rceil$. If the SDP is feasible, then we simply run the rounding algorithm from Theorem 3. This ensures that we find a biclique of size at least $(\log n)^{10}$, which yields $n/(\log n)^{10} \leq O\left(n \cdot \left(\frac{\log\log n}{\log n}\right)^3\right)$-approximation.

If the SDP is infeasible, then the maximum balanced biclique has size at most $O\left(n \cdot \frac{\log\log n}{\log n}\right)$. We run Chalermsook et al.'s algorithm (Theorem 2). If the maximum balanced biclique has size at least $n/\log^3 n$, we have $t \leq \log^3 n$ and thus the algorithm produces $\Theta\left(\left(\frac{\log n}{\log\log n}\right)^2\right)$-biclique, which yields $O\left(n \cdot \left(\frac{\log\log n}{\log n}\right)^3\right)$-approximation. On the other hand, if the maximum balanced biclique has size at most $n/\log^3 n$, we automatically get $n/\log^3 n \leq O\left(n \cdot \left(\frac{\log\log n}{\log n}\right)^3\right)$-approximation. ◀

5 Conclusion and Open Questions

In this work, we give a polynomial-time $\left(\frac{n}{\widetilde{\Omega}((\log n)^3)}\right)$-approximation algorithm for the Maximum Balanced Biclique problem. Our algorithm is via SDP rounding in the case where the optimum is large, combined with the known result of Chalermsook et al. [8] in the case where the optimum is small. Apart from trying to improve the approximation ratio further, it remains an interesting question whether the former case can be handled directly via combinatorial algorithms. We remark that, in the Maximum Clique case, subgraph removal techniques–which are completely combinatorial–suffice for this case [7].

Another intriguing question is whether one can achieve a similar approximation for the Maximum *Edge* Biclique (MEB) problem. In MEB, we are looking for a (not necessarily balanced) biclique in G that contains the maximum number of edges. A trivial algorithm yields n-approximation, and a simple "enumeration style" algorithm can improve this to $\frac{n}{\widetilde{\Omega}(\log n)}$. However, we are not aware of any further improvement to this. In particular, it is unclear how to apply SDP rounding when the optimum is large.

References

1 Noga Alon and Nabil Kahalé. Approximating the independence number via the theta-function. *Math. Program.*, 80:253–264, 1998. doi:10.1007/BF01581168.

2 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998. doi:10.1145/278298.278306.

3 Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *J. ACM*, 45(1):70–122, 1998. doi:10.1145/273865.273901.

4 Maria Axenovich, Jean-Sébastien Sereni, Richard Snyder, and Lea Weber. Bipartite independence number in graphs with bounded maximum degree. *SIAM J. Discret. Math.*, 35(2):1136–1148, 2021. doi:10.1137/20M1321760.

5 Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, pcps, and nonapproximability-towards tight results. *SIAM J. Comput.*, 27(3):804–915, 1998. doi:10.1137/S0097539796302531.

6 Amey Bhangale, Rajiv Gandhi, Mohammad Taghi Hajiaghayi, Rohit Khandekar, and Guy Kortsarz. Bi-covering: Covering edges with two small subsets of vertices. *SIAM J. Discret. Math.*, 31(4):2626–2646, 2017. doi:10.1137/16M1082421.

7 Ravi B. Boppana and Magnús M. Halldórsson. Approximating maximum independent sets by excluding subgraphs. In John R. Gilbert and Rolf G. Karlsson, editors, *SWAT*, pages 13–25, 1990. doi:10.1007/3-540-52846-6_74.

8 Parinya Chalermsook, Wanchote Po Jiamjitrak, and Ly Orgo. On finding balanced bicliques via matchings. In *WG 2020*, pages 238–247, 2020. doi:10.1007/978-3-030-60440-0_19.

9 Lars Engebretsen and Jonas Holmerin. Towards optimal lower bounds for clique and chromatic number. *Theor. Comput. Sci.*, 299(1-3):537–584, 2003. doi:10.1016/S0304-3975(02)00535-2.

10 Uriel Feige. Relations between average case complexity and approximation complexity. In *STOC*, pages 534–543, New York, NY, USA, 2002. ACM. doi:10.1145/509907.509985.

11 Uriel Feige. Approximating maximum clique by removing subgraphs. *SIAM J. Discret. Math.*, 18(2):219–225, 2004. doi:10.1137/S089548010240415X.

12 Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *J. ACM*, 43(2):268–292, 1996. doi:10.1145/226643.226652.

13 Uriel Feige and Shimon Kogan. Hardness of approximation of the balanced complete bipartite subgraph problem. Technical report, Weizmann Institute of Science, Rehovot, Israel, 2004.

14 Uriel Feige and Shimon Kogan. Balanced coloring of bipartite graphs. *J. Graph Theory*, 64(4):277–291, 2010. doi:10.1002/JGT.20456.

15 William Feller. *An introduction to probability theory and its applications, Volume 2*, volume 2. John Wiley & Sons, 1991.

16 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

17 Eran Halperin. Improved approximation algorithms for the vertex cover problem in graphs and hypergraphs. *SIAM J. Comput.*, 31(5):1608–1623, 2002. doi:10.1137/S0097539700381097.

18 Johan Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. In *FOCS*, pages 627–636, 1996. doi:10.1109/SFCS.1996.548522.

19 David S. Johnson. The np-completeness column: An ongoing guide. *J. Algorithms*, 8(5):438–448, September 1987. doi:10.1016/0196-6774(87)90021-6.

20 Richard M. Karp. Reducibility among combinatorial problems. In *Proceedings of a symposium on the Complexity of Computer Computations*, pages 85–103, 1972. doi:10.1007/978-1-4684-2001-2_9.

21 Subhash Khot. Improved inaproximability results for maxclique, chromatic number and approximate graph coloring. In *FOCS*, pages 600–609, 2001. doi:10.1109/SFCS.2001.959936.

22 Subhash Khot. Ruling out ptas for graph min-bisection, dense k-subgraph, and bipartite clique. *SIAM Journal on Computing*, 36(4):1025–1071, 2006. doi:10.1137/S0097539705447037.

23 Subhash Khot and Ashok Kumar Ponnuswami. Better inapproximability results for maxclique, chromatic number and min-3lin-deletion. In *ICALP*, pages 226–237, 2006. doi:10.1007/11786986_21.

24 Jon M. Kleinberg and Michel X. Goemans. The lovász theta function and a semidefinite programming relaxation of vertex cover. *SIAM J. Discret. Math.*, 11(2):196–204, 1998. doi:10.1137/S0895480195287541.

25 László Lovász. On the shannon capacity of a graph. *IEEE Trans. Inf. Theory*, 25(1):1–7, 1979. doi:10.1109/TIT.1979.1055985.

26 Pasin Manurangsi. Almost-polynomial ratio eth-hardness of approximating densest k-subgraph. In *STOC*, pages 954–961, 2017. doi:10.1145/3055399.3055412.

27 Pasin Manurangsi. Inapproximability of maximum biclique problems, minimum k-cut and densest at-least-k-subgraph from the small set expansion hypothesis. *Algorithms*, 11(1):10, 2018. doi:10.3390/A11010010.

28 Pasin Manurangsi. Answer to "approximation algorithm for balanced bipartite independent set?". Theoretical Computer Science Stack Exchange, August 2022. Online; accessed 14-April-2026. URL: https://cstheory.stackexchange.com/a/51812.

29 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory Comput.*, 3(1):103–128, 2007. doi:10.4086/TOC.2007.V003A006.

Stochastic Caching via Subset Entropy

Ravi Kumar
Google Research, Mountain View, CA, USA

Roie Levin
Rutgers University, Piscataway, NJ, USA

Joseph (Seffi) Naor
Technion – Israel Institute of Technology, Haifa, Israel

Debmalya Panigrahi
Duke University, Durham, NC, USA

Abstract

A classic approach to beyond worst-case algorithm design is to impose stochastic assumptions on the input. However, a limiting feature of such stochastic analyses is that, by the min-max principle, performance on worst-case distributions mirrors that of randomized algorithms on worst-case inputs. In other words, the same shortcoming of worst-case analysis – its inability to distinguish between "easy" and "hard" instances – reappears as an inability to distinguish between "easy" and "hard" distributions. This raises a natural question: *Can we characterize "easy" input distributions with useful beyond worst-case bounds?*

A canonical example is the *stochastic caching* problem (Aho, Denning, and Ullman, 1971). When the page requests are drawn i.i.d. from the uniform distribution, the best achievable competitive ratio is $O(\log k)$, matching the performance of the best randomized algorithm on worst-case instances (Fiat, Karp, Luby, McGeoch, Sleator, and Young, 1991). However, when the input distribution has less entropy, intuition suggests that we should be able to do better by exploiting the information provided by the distribution. In this paper, we formalize this intuition by defining a new information-theoretic parameter of probability distributions called *subset entropy*. We then use this parameter to give a fine-grained characterization of the competitive ratio of stochastic caching, including a new analysis for the well-known LRU algorithm on stochastic inputs.

While our technical results are for the caching problem, we believe the broader principle – parameterizing algorithmic performance by an entropy measure of the input – is of independent interest and might apply to other online and stochastic optimization problems. Indeed, for many other fundamental problems such as (comparison-based) sorting, online matching, online load balancing, etc., the hardest stochastic instances involve high-entropy distributions. We hope our work is a step toward a broader theory of fine-grained algorithmic performance for this class of problems.

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases Online Algorithms, Beyond Worst-Case Analysis, Caching, Paging, Entropy

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.2

Category APPROX

Related Version *Arxiv version*: https://arxiv.org/abs/2607.03947

Funding *Joseph (Seffi) Naor*: Supported in part by ISF grant 3001/24 and United States – Israel BSF grant 2022418.
Debmalya Panigrahi: Supported in part by NSF grants CCF-2329230 and CCF-1955703.



© Ravi Kumar, Roie Levin, Joseph Naor, and Debmalya Panigrahi;
licensed under Creative Commons License CC-BY 4.0

1 Introduction

Stochastic optimization is a well-known and extensively-studied framework for beyond worst-case analysis of algorithmic problems. The basic premise is to make an assumption about the input being drawn from a probability distribution, rather than being generated adversarially, thereby enabling algorithms that exploit structural properties of the distribution. However, a common shortfall of this strategy is that algorithmic performance on the worst-case input distribution often mirrors that of the best randomized algorithm for worst-case inputs. This equivalence, a consequence of the min-max principle (Yao [35]), is often useful for proving randomized lower bounds for algorithms but, conversely, also lower bounds the performance of the best stochastic algorithms. This is rather limiting because it brings back the main deficiency of worst-case analysis – its inability to separate "easy" from "hard" instances – as being unable to differentiate between "easy" and "hard" distributions. This raises the natural question: *can we obtain a finer-grained characterization of input distributions that yields beyond worst-case bounds for "easy" distributions?*

The Caching Problem. A case in point is the classical online *caching* (or paging) problem. In this problem, a sequence of page requests arrives online from a universe of n pages. The goal is to serve each request using a cache that can only hold k ($\ll n$) pages at a time while minimizing the total number of page swaps. Caching is used in almost every extant computer system. It is also a focal point for modern developments in online algorithms such as the online primal-dual method, projections, and mirror descent [13, 12, 11]. The problem is considered well-understood in online adversarial models: in their seminal work, Sleator and Tarjan [33] showed that the *least-recently used* (LRU) policy is k-competitive, and that any deterministic online algorithm is at least k-competitive. When randomization is allowed, Fiat et al. [17] showed an elegant and improved $O(\log k)$-competitive algorithm. Unfortunately, this bound is asymptotically tight for randomized algorithms even when page requests are sampled i.i.d. from a known distribution, since no algorithm can do better on the uniform distribution over $n = k + 1$ pages [17].

Yet, our understanding of the caching problem is wanting even in the simplest "beyond worst-case" setting, where each request is drawn independently from a fixed distribution. As noted, the uniform distribution *is* the worst-case distribution – but this is also the least compelling case for caching. If requests are uniform, there is no reason for a caching algorithm to prefer one page over another, and any caching algorithm is as good as any other. In contrast, if page requests are skewed toward a set of fewer than k pages, smart caching can be effective, as one can hope to identify the frequently requested pages and prioritize evicting others.

This leads to our main questions: *Which distributions admit better-than-worst-case competitive ratios? And which online caching algorithms attain tighter bounds for distributions that are caching-friendly?*

1.1 Our Results

We study the *stochastic caching* problem, where we assume the input sequence of requests is drawn independently from a fixed distribution. The *competitive ratio* of a stochastic caching algorithm is the ratio of the expected cost of the algorithm and the expected cost of an optimal offline solution, where both expectations are taken over the input distribution (and the algorithm, if randomized).

Our first contribution is a new measure called *subset entropy* to quantify the competitive ratio of a stochastic caching instance. A reasonable first guess for such a measure would be the *entropy* of the input distribution; indeed, this would align in the case of the uniform distribution over $n = k + 1$ pages, where the entropy is $\Theta(\log k)$. However, if the distribution places $1-\varepsilon$ mass on one page, and $\varepsilon/(n-1)$ on each of the remaining pages, then the entropy is near zero, but this is essentially the same stochastic caching instance as before, but with a cache of size $k-1$, and with the $n-1$ pages of low probability in the original instance.

This motivates our definition of subset entropy for a distribution $\mathcal{D}$: it is the maximum, taken over all subsets in the support of a distribution $\mathcal{D}$, of the entropy of $\mathcal{D}$ conditioned to the subset. In the case of stochastic caching, we only need to consider subsets of size at most k in the support; we call this *k-subset entropy* and denote it $\mathscr{H}(\mathcal{D}, k)$, or $\mathscr{H}$ for short when $\mathcal{D}$ and k are clear from context. In Section 3, we show how to achieve competitive ratio $O(\mathscr{H})$ for every input distribution $\mathcal{D}$.[1]

▶ **Theorem 1** (Informal). *There is a caching algorithm that for a given distribution $\mathcal{D}$ and a sequence of page requests drawn i.i.d. from $\mathcal{D}$, achieves a competitive ratio of $O(\mathscr{H})$.*

Note that $\mathscr{H}$ is at most the worst case bound $O(\log k)$, but it is also much sharper for many distributions. In fact, we show in Section 2 that the only distributions for which $\mathscr{H} = \Theta(\log k)$ are those that are nearly uniform, namely those for which some subset of $\Omega(\text{poly}(k))$ pages have probabilities within a constant factor of each other. To our knowledge, the notion of subset entropy is novel and would be interesting to explore in other contexts as well. For example, does it encapsulate interesting information-theoretic properties of a distribution? We show one property in Section 2: informally, we show that it captures the cardinality of the largest (approximately) uniform distribution embedded in the overall distribution. A different question: is $\mathscr{H}$ (exactly) computable efficiently? We answer this in the affirmative in Appendix E, given explicit access to the distribution. We hope this new parameter will be studied from other angles in the future.

The algorithm we use to establish Theorem 1 is natural: since $\mathcal{D}$ is known, we keep the top k-1 pages by probability in cache permanently and use the last cache slot to serve all other requests;[2] we call this algorithm Top-k. Indeed, in an important work from 1971, Aho, Delling, and Ullman [2] showed that this is the *optimal* online caching strategy in the i.i.d. setting.[3] Though the algorithm is unsurprising, the analysis is new. We show that Top-k is $O(\mathscr{H})$-competitive against the *offline* optimum. This quantifies the precise role of randomness in the input sequence on the competitive ratio, namely the difference between knowing the input *distribution* versus the actual *realization* of the input sequence.

The previous Top-k algorithm requires knowing the input distribution (we call this the *known* i.i.d. setting). Now, we consider the *unknown* i.i.d. setting where the caching algorithm no longer has access to the input distribution. Intuitively, we would still like to keep the top k pages by probability in the cache. Ordering pages by decreasing probability is equivalent to ordering them by increasing expected length of the interval between consecutive requests. Since we do not have access to the expected interval lengths, we use as a surrogate the length

[1] There do exist distributions for which $\mathscr{H}$ is large and yet the distribution admits smaller competitive ratio. E.g., the uniform distribution with $n = 10k$ pages has $\mathscr{H} = O(\log k)$ but any algorithm is constant-competitive. To our knowledge this is true of most parameterized results, since it is rare to identify a parameter that guarantees *both* upper and lower bounds.

[2] For ease of analysis, we define the Top-k algorithm slightly differently from the optimal online strategy; see Section 3.

[3] In the stochastic setting, an optimal online solution is well-defined. It can be obtained from a Markov decision process via a dynamic program which, in general, can have an exponential-sized state space.

of the interval since the last request for every page – i.e., keep in cache the k most recently requested pages. This is none other than the well-known LRU algorithm, and our second result shows that it is $O(\mathscr{H}^3)$-competitive even in the unknown i.i.d. setting.

▶ **Theorem 2.** *When given a sequence of page requests drawn i.i.d. from an unknown distribution $\mathcal{D}$, the* LRU *caching algorithm achieves a competitive ratio of $O(\mathscr{H}^3)$.*

As intuition would predict, LRU approximately learns which pages in cache are most likely to be requested furthest in future and evicts these first. How well it does so depends on how predictable is this information, which in turn is captured by the subset entropy of $\mathcal{D}$. Since $\mathscr{H} = O(\log k)$, this also shows that LRU is $O(\log^3 k)$-competitive for unknown i.i.d. inputs. To our knowledge, this is the first analysis of LRU in a stochastic setting that yields an $o(k)$-competitive factor.

A more precise (although also not as lightweight) alternative to LRU is the *least-frequently used* (LFU) algorithm that always evicts the page with the least number of requests so far. Our third result shows that LFU is $O(\mathscr{H})$-competitive for i.i.d. inputs. This implies that up to constant factors, the known and unknown i.i.d. settings have the same competitive ratio.

▶ **Theorem 3.** *When given a sequence of page requests drawn i.i.d. from an unknown distribution $\mathcal{D}$, the LFU caching algorithm achieves a competitive ratio of $O(\mathscr{H})$.*

This theorem is not surprising in an asymptotic sense, since LFU eventually converges to the Top-k algorithm. But a priori, the rate of convergence could be arbitrarily slow depending on the difference between the probability values of pages requests. We show that the additive loss of the LFU algorithm vis-à-vis the Top-k algorithm is $O(k\mathscr{H})$, and is therefore absorbed in the multiplicative factor in Theorem 3.[4] In other words, LFU matches the performance of Top-k *even when it has not finished learning the input distribution.* We view this as an interesting insight into learning-augmented algorithms: it demonstrates the difference between statistically learning a distribution and learning it well enough to devise an effective algorithmic strategy.

1.2 Techniques and Overview

We start in Section 2 with our characterization of k-subset entropy in terms of the largest (up to size k) approximately uniform distribution embedded in $\mathcal{D}$. Indeed, by definition, any conditional distribution in $\mathcal{D}$ gives a lower bound on subset entropy. To show the converse, we use a factor-revealing LP to bound the maximum value of subset entropy when every approximately uniform distribution is size-constrained.

In Section 3, we show that the Top-k algorithm has a competitive ratio of $O(\mathscr{H})$, thereby establishing Theorem 1. The k pages with the largest values of p_i are permanently in the cache in the Top-k algorithm; we call them the *head* of the distribution and call the remaining pages the *tail*. The expected cost of the Top-k algorithm is immediate: it is the total probability in the tail, denoted $p(\mathbb{T})$. But how do we lower bound the expected cost of an offline optimal solution? The usual technique is to use dual fitting, i.e., write primal and dual LPs and then construct a feasible dual solution of sufficiently large value. Here, since the instance itself is random, so are the coefficients of the primal and dual LPs, and what we really have is a *distribution* over LPs. How do we argue about the expected value of the optimal solution of such a distribution? (Contrast this to randomized dual fitting [15] where the LPs are fixed but the primal and dual solutions are random.)

[4] k is required for all algorithms as an initialization cost from an empty cache.

We start by writing the "expected" primal-dual LPs whose coefficients are the expected values of the (random) coefficients over the distributions of the primal and dual LPs. For this LP, we construct a feasible dual solution of value $p(\mathbb{T})$ that satisfies these "expected constraints". Interestingly, this dual solution to the "expected" LP turns out to be quite different from those used in standard primal-dual analysis of caching algorithms (e.g., in [5]). However, we want the dual solution to be feasible for the actual realized constraints, not the expected constraints. Constraints for which the realized dual coefficients are "close" to their expected values (in a specific sense that we define later) are already approximately satisfied by the expected dual. The main challenge is to handle the "tail constraints", namely those that realize very differently from their expected form. These realizations can happen in many different ways and tracking their feasibility precisely is onerous. Instead, we carefully zero out dual variables to make sure that the tail constraints are also satisfied. This creates a distribution over dual solutions, since the set of tail constraints, and therefore, zeroed out variables, depends on the particular realization. Nevertheless, crucially, for each realized dual instance, the corresponding dual is *always* feasible. The key is to show that since each constraint is a tail constraint with small probability, we do not lose too much in the objective by zeroing out dual variables. In fact, we precisely lose a factor of $O(\mathscr{H})$. Hence this process gives a randomly defined dual solution that is deterministically feasible for the corresponding random instance, and the expected value of the dual objective over these random solutions is $\Omega\left(p(\mathbb{T})/\mathscr{H}\right)$.

In Section 4, we study the LRU algorithm and establish Theorem 2. Recall that LRU is an empirical implementation of the Top-k algorithm which evicts pages based on the *expected* length time between subsequent requests. Thus it is tempting to imagine that the ideal scenario for LRU is when the empirical length of a request interval exactly matches its expected value for every page. Unfortunately, this intuition is misleading. Consider a uniform input distribution $\mathcal{D}$ over $n = k + 1$ pages. If an instantiation creates a round-robin request sequence, then the empirical length of a request interval is exactly $k + 1$, i.e., its expected value. However, LRU is k-competitive for this input sequence! Therefore, we need to use the symmetry-breaking properties of a random sequence in our analysis. Suppose a page i is requested at time t. If page i is not in the cache at time t, then by the properties of LRU, page i is not among the last k distinct pages requested before time t. What is the probability of this event? By reversing time, this is the same as the probability that page i is not going to be among the next k distinct pages to be requested after time t. Now, define a random variable for every page denoting the time till the first request of the page. Although these times are dependent, to first approximation, we can imagine that each page i is running an independent Poisson process with rate p_i, i.e., we can approximate the length of the request intervals by exponential variables. Then, we use the properties of Poisson processes to bound the probability that at least k other processes fired before process i, eventually proving Theorem 2.

In Section 5, we study the LFU algorithm and prove Theorem 3. We note that LFU might suffer more cache misses than Top-k because of inversions in the ranked order of pages by empirical frequency counts vis-à-vis their true probabilities. We bound the effect of these inversions in two parts. First, we show that inversions due to pairs that have probabilities within (say) a factor of 2 only incur an expected cost $O(p(\mathbb{T}))$. It is crucial that these inversions do not cost us in the additive term because such inversions might persist for an arbitrarily long time (e.g., if two pages differ in probability by ε). It remains to bound the cost due to inversions where one page is twice as likely as the other. We show that after $O(\mathscr{H})$ requests for a page, its count will forever be higher (with good probability) than the count of any page with less than half its probability. This incurs an additive loss of $O(k\mathscr{H})$ on the pages in the head, which is ultimately absorbed in the multiplicative factor.

The Bigger Picture. Having discussed our technical results for stochastic caching, we now return to the broader perspective: obtaining finer-grained analyses for stochastic optimization problems. We argue that identifying a distributional quantity, such as entropy, and analyzing algorithms with respect to this quantity, offers a promising direction for beyond worst-case analysis in stochastic settings. Indeed, for many fundamental problems – such as comparison-based sorting, online matching, online load balancing, and related tasks – the hardest stochastic instances correspond to high-entropy (often uniform) distributions. Intuitively, this is consistent with the view that distributions with higher entropy are less informative than those with lower entropy. We view our work as a first step toward a theory, based in entropy and its variants, that provides a refined analysis of algorithmic performance across stochastic versions of this rich class of problems.

1.3 Related Work

In a classic paper, Belady [8] introduced the optimal "furthest in the future" rule for evicting pages from a cache in the offline setting. Soon after, stochastic versions of the online caching problem were considered and adaptations of Belady's rule to online distributional settings were proposed and analyzed [2, 18]. Over the next few decades, a rich array of input models were introduced for online caching; some highlights include the work of Lund et al. [26] and Pabbaraju and Vakilian [29], who obtained approximations against the best online strategy even in partial information settings, the Markov paging model of Karlin et al. [23], where the input sequence is a Markov chain, and the access graph setting of Borodin et al. [10] (see also [9, Ch. 5]), where the request sequence is generated by a simple walk on an "access graph" defined on the pages. Somewhat related to our work is that of Pandurangan and Upfal [31], who studied the relationship between the performance of the best online caching algorithm and the entropy of a request sequence generated by a stochastic process. In particular, they showed that entropy is not sufficient to capture the performance of online caching algorithms.

We note that many further variants of online caching have been proposed, e.g., when pages have non-uniform weights [4, 21, 37], non-uniform sizes [3, 5, 22], as well as non-standard caching models such as elastic caches [19], caching with time windows [20], and caching with dynamic weights [16], interleaved caching [25], and caching with machine learning predictions [27, 6]. In contrast to the i.i.d. setting that we study in this paper, many models try to capture *locality of reference*, i.e., the phenomenon that pages requested recently tend to be re-requested [24, 36, 7, 30]. The validity of locality of reference varies by application domain, e.g., is more applicable for centralized settings like operating systems on computing devices compared to distributed and resource-shared settings like data centers where the requests come from a large number of independent processes. Nevertheless, we view the extensive literature on stochastic paging – including our work – and that on models capturing locality of reference as complementary, since they capture different aspects of a fundamental problem. Indeed, this complementarity can sometimes be leveraged to obtain richer models that try to incorporate both phenomenon, such as Markov paging [23].

In our work, the input model is the classic i.i.d. setting, but we are interested in characterizing input distributions that are amenable to caching and obtaining better-than-worst-case bounds for such inputs. In this sense, it aligns with the recent focus on "beyond worst-case" algorithm design, and in particular, with the emerging field of learning-augmented algorithms, e.g., [1]. The main difference is that while the typical goal in the latter context is to use additional information about the input and automatically adapt the performance bounds to the *quality* of this information, our goal is to adapt performance bounds to the quality of the input distribution itself without using additional information about the input. We point the reader to an excellent survey of "beyond worst-case" caching models in [32, Ch. 24] for a detailed discussion of related models.

2 Problem Definition and Background

2.1 The Stochastic Caching Problem

There is a universe of n pages indexed $[n] = \{1, \ldots, n\}$, a cache of size $k \le n$, a request sequence of length T indexed by $[T]$, and a distribution $\mathcal{D}$ on $[n]$. At each time $t \in [T]$, the algorithm gets a page request $r_t \in [n]$, where r_t is drawn independently from $\mathcal{D}$. The algorithm maintains at most k pages in the cache at all times with the constraint that page r_t must be in the cache at time t. Swapping in/out a page in the cache incurs unit cost; the goal is to minimize the total cost, i.e., the number of page swaps. We call this the *stochastic caching* problem.

The distribution $\mathcal{D}$ is defined by probabilities $p_1, \ldots, p_n$, where $p_i > 0$ is the probability that page $i \in [n]$ is requested at any given time. Clearly, $\sum_i p_i = 1$. We also denote $p(S) = \sum_{i \in S} p_i$ for a subset $S \subseteq [n]$. Wlog, we index pages in decreasing order of probabilities, i.e., $p_1 \ge p_2 \ge \cdots \ge p_n$, breaking ties arbitrarily. The top k pages are called the *head* of the distribution $\mathcal{D}$ and denoted $\mathbb{H} := [k]$; the remaining pages $\mathbb{T} := [n] \setminus [k]$ comprise the *tail* of the distribution. The total probability mass in the tail is denoted $p(\mathbb{T}) := \sum_{i=k+1}^{n} p_i = 1 - p(\mathbb{H})$. In this paper, we consider two cases: when $\mathcal{D}$ is *known* and when $\mathcal{D}$ is *unknown* to the algorithm.

2.2 Subset Entropy

For a distribution $\mathcal{D}$ with support U and for a subset $S \subseteq U$, let $\mathcal{D}_S$ denote the distribution $\mathcal{D}$ conditioned on S, i.e., $\mathcal{D}_S(u) = \frac{\mathcal{D}(u)}{\sum_{u \in S} \mathcal{D}(u)}$ for $u \in S$. Let $\mathrm{H}(\cdot)$ denote the *Shannon entropy* function: $\mathrm{H}(\mathcal{D}_S) := -\sum_{u \in S} \mathcal{D}_S(u) \cdot \log \mathcal{D}_S(u)$. (All logarithms throughout the paper are base 2 unless otherwise stated.)

We define *k-subset entropy* of a distribution $\mathcal{D}$ as follows:

▶ **Definition 4** (k-Subset Entropy)**.** For a discrete distribution $\mathcal{D} : U \to [0, 1]$ and any positive integer $k \le |U|$, the *k-subset entropy* of $\mathcal{D}$ is defined as the following:

$$\mathscr{H}(\mathcal{D}, k) = \max_{S \subseteq U : |S| \le k} \mathrm{H}(\mathcal{D}_S). \tag{2.1}$$

While k-subset entropy suffices in the context of caching, an interesting information-theoretic counterpart is the unrestricted version that maximizes entropy of conditional distributions over *all* subsets of U. We call this *subset entropy*, and it is the special case of k-subset entropy when $k = n$.

Next, we prove an important fact about k-subset entropy, which we rely on in the rest of the paper.

▶ **Lemma 5** (k-Subset Entropy Lemma)**.** *Suppose we have a distribution $\mathcal{D} : U \to [0, 1]$. Let $S_\ell = \{u \in U : 1/2^{\ell+1} < \mathcal{D}(u) \le 1/2^{\ell}\}$ for any $\ell \ge 1$ and let $n_{\max} := \max_\ell |S_\ell|$. Then,*

$$\mathscr{H}(\mathcal{D}, k) = \Theta(\log(\min\{k, n_{\max}\})).$$

One direction is immediate: clearly $\mathscr{H}(\mathcal{D}, k) \ge \log(\min(k, n_{\max}))$ since the set S in the definition (2.1) could be chosen as (a subset of size $\le k$ of) the S_ℓ for which $n_{\max} = |S_\ell|$, and by the definition of Shannon entropy, $H(D_{S_\ell}) = \Theta(\log |S_\ell|)$. Surprisingly, the converse holds: if $\mathscr{H}(\mathcal{D}, k) = h$, then there is a set $S \subseteq U, |S| \ge 2^{\Theta(h)}$, over which the conditional distribution $\mathcal{D}_S$ is uniform up to constants. Intuitively, this means that the k-subset entropy of a distribution is dictated by the largest (up to k) approximately uniform distribution

embedded in it. The proof (Appendix A) is via dual fitting of a factor revealing LP: we show that even if we ignore the original distribution $\mathcal{D}$ and choose the probabilities of the pages in the maximizing set S completely freely, if we are constrained to choosing at most $n_{\max}$ pages within a factor 2 of each other, we cannot give S a conditional entropy greater than $O(\log n_{\max})$.

In Appendix E, we also show that the maximizing set in the Definition (2.1) is always a subinterval of U, when arranged in sorted order of probabilities. This gives an efficient algorithm for computing k-subset entropy *exactly*. Nevertheless, the approximate characterization above is convenient to work with in our applications.

We use henceforth the shorthand $\mathscr{H}$ to denote the k-subset entropy $\mathscr{H}(\mathcal{D}, k)$, since it is implicit that the distribution $\mathcal{D}$ is the input distribution and the parameter k is the cache size.

3 Stochastic Caching for a Known Distribution

We begin with the known distribution setting. Recall our simple algorithm Top-k: keep the pages of the head $\mathbb{H}$ in cache; when a page from the tail $\mathbb{T}$ is requested, swap out an arbitrary page in cache for the requested page, then immediately swap the same page back. The main result of this section, which establishes Theorem 1, is the following:

▶ **Theorem 6.** *The competitive ratio of the algorithm* Top-k *for the stochastic caching problem for input distribution $\mathcal{D}$ is $O(\mathscr{H})$.*

To establish Theorem 6, we first bound the expected cost of the Top-k algorithm.

▶ **Lemma 7.** *The expected cost of the* Top-k *algorithm for each request is $2p(\mathbb{T})$.*

Proof. For each request, the probability that the requested page is not in the cache is exactly $p(\mathbb{T})$, in which case the algorithm performs two evictions. ◀

The harder challenge is to lower bound the expected cost of an offline optimal solution. To this end, we fit a dual to the LP relaxation of the knapsack-cover (or KC) integer programming formulation for caching (see, e.g., [5])[5]. As noted in the introduction, while focusing on LP duality is a reasonable instinct, the main challenge is that the LP is defined via a distribution over coefficients, i.e., both the primal and dual LPs are random. We have to produce a feasible dual solution for *every* realization of the random instance, and then bound the expected objective value across the realizations.

Fortunately, since we are in an i.i.d. setting, the distribution over dual constraints that apply at any given time t is time-invariant, i.e., is independent of t. We view this distribution over dual constraints in two parts – the set of constraints that apply within a high probability interval, and the remaining constraints that are in the tail of the distribution. For the former, we set dual variables corresponding to the "expected" constraints and these are approximately feasible. For the latter, there are a large number of possibilities and it is onerous to track feasibility across all these realizations. Instead, we carefully zero out a subset of dual variables to ensure feasibility for the constraints in the tail of the distribution. Of course, the set of dual constraints that fall in the tail region is random, and therefore the set of zeroed out variables is also random. Nevertheless, we show that since a constraint

[5] Note that our choice of the KC inequality strengthened formulation is for convenience; it is not strictly necessary because the standard LP formulation is already integral.

is in the tail with low probability (by standard concentration inequalities), this last step of zeroing out variables to ensure feasibility has limited impact on the expected value of the dual objective.

▶ **Lemma 8.** *The expected cost of an offline optimal solution is* $\Omega\left(\frac{p(\mathbb{T})}{\mathscr{H}}\right)$.

Proof Sketch. Let $x(i,r)$ be the indicator variable for the event that page i is evicted in the time interval $I(i,r) := [t(i,r)+1, t(i,r+1)-1]$, where $t(i,r)$ is the time of the rth request to page i. Let $\rho(i,t)$ be the number of requests to page i up to and including time t. Consider the formulation:

$\min \sum_{i=1}^{n} \sum_{r=1}^{\rho(i,T)} x(i,r)$	subject to
$\sum_{i \in S \setminus \{r_t\}} x(i, \rho(i,t)) \geq \lvert S \rvert - k$	$\forall t \in [T]$ $\forall S \subseteq [n]$
$x(i,r) \geq 0$	$\forall i \in [n]$ $\forall r \in [\rho(i,T)]$

$\max \sum_{t=1}^{T} \sum_{S \subseteq [n]} (\lvert S \rvert - k) \cdot y_S^t$	subject to
$\sum_{t \in I(i,r)} \sum_{S: i \in S} y_S^t \leq 1$	$\forall i \in [n]$ $\forall r \in [\rho(i,T)]$
$y_S^t \geq 0$	$\forall t \in [T]$ $\forall S \subseteq [n]$

This formulation is standard, but our dual fitting is not (see [5] for the usual way the online dual is fit using multiplicative weights). We start by finding a stationary solution to the *expected* dual, i.e., find a vector y that is the same for each $t \in [T]$, such that the expected dual constraint is satisfied.

Intuitively, we should expect every time t to lie in an interval of length $1/p_i$ between subsequent requests to page i, and hence these expected constraints would ask that there be less than 1 total y mass on page subsets containing i on these intervals, i.e., that there be less than p_i mass on these subsets per time step t. Define

$$y_S^t = \begin{cases} p_i - p_{i+1} & \text{if } S = [i] \text{ and } i \in \mathbb{T} \setminus \{n\}, \\ p_n & \text{if } S = [n], \\ 0 & \text{otherwise.} \end{cases}$$

Then, by design, for all $i \in \mathbb{T}$ we have $\sum_{S: i \in S} y_S^t = \sum_{j=i}^{n-1} (p_j - p_{j+1}) + p_n = p_i$, as desired. For the remaining $i \in \mathbb{H}$, we have $\sum_{S: i \in S} y_S^t = \sum_{j=k+1}^{n-1} (p_j - p_{j+1}) + p_n = p_{k+1} \leq p_i$, Furthermore, the dual objective (per time step) is $\sum_S (\lvert S \rvert - k) \cdot y_S^t = n \cdot p_n + \sum_{j=1}^{n} j \cdot (p_j - p_{j+1}) = \sum_{i \in \mathbb{T}} p_i$.

However, this is too good to be true! If this were a legal dual solution, weak duality would imply that Top-k in fact achieves competitive ratio $O(1)$, which we know is impossible, as demonstrated by the uniform distribution case. Indeed, for the dual to witness a true lower bound on the primal program, it must be feasible with probability 1, and our stationary solution, as described, might violate dual constraints corresponding to intervals that happen to be longer than their expectation.

To construct our actual dual, we start with the stationary y described above. We then show how to carefully drop some coordinates to account for the random fluctuation in interval lengths. Since the dual is a packing problem, we can always make a solution feasible this way. The crux is to do this while approximately preserving the objective value. In Appendix B, we give the details of our modified dual construction and show that we only lose an $O(\mathscr{H})$ factor in the objective. ◀

4 The LRU Algorithm

In this section, we analyze the popular *least recently used* (LRU) algorithm for stochastic caching. Note that LRU does not use distributional information, and hence can be used in the unknown distribution setting. Our main result is:

▶ **Theorem 9.** *The* LRU *algorithm has a competitive ratio* $O(\mathscr{H}^3)$ *for the stochastic caching problem.*

We now provide the main ideas needed for proving this and defer the technical claims to Appendix C. We saw in Section 3 that the expected cost of an offline optimal solution is $\Omega\left(\frac{p(\mathbb{T})}{\mathscr{H}}\right)$. Hence, it suffices to bound the expected cost of LRU as $O(\mathscr{H}^2 \cdot p(\mathbb{T}))$. Note that LRU has a cache miss when the requested page is not among the last k distinct pages requested. Thus, for any page i, the probability that i incurs a cache miss (at any point of time) is the same as that of page i not being among the first k distinct pages requested in a sequence of pages drawn from $\mathcal{D}$. We denote this probability q_i for page i. With this notation, the expected cost of the LRU algorithm for any request is $\sum_{i\in[n]} p_i q_i$. Our goal from here on is to show

$$\sum_{i\in[n]} p_i q_i \le O(\mathscr{H}^2 \cdot p(\mathbb{T})). \tag{4.2}$$

Before we begin, note that when $p(\mathbb{T}) > 1/2$, any algorithm, even one that incurs a cache miss at every time step, has cost $O(p(\mathbb{T}))$. Thus, we can assume without loss of generality that:

▶ **Assumption 10.** *The tail has total probability* $p(\mathbb{T}) \le 1/2$.

Note that the contribution of the tail to the LHS of (4.2) is easy to handle, since the q_i's are probabilities.

▶ **Observation 11** (Bounding $\mathbb{T}$)**.** *The contribution of* $\mathbb{T}$ *is bounded as* $\sum_{i\in\mathbb{T}} p_i q_i \le p(\mathbb{T})$.

This leaves us with the task of bounding the contribution of the head $\mathbb{H}$ to the LHS of (4.2).

4.1 Probability Levels

We first group the pages in the head $\mathbb{H}$ into doubling intervals by probability values p_i. In particular, for $\ell \ge 0$, we use L_ℓ to denote the set of all pages i with $p_i \in [1/2^{\ell+1}, 1/2^\ell)$; we say that the pages in p_ℓ are at level ℓ and denote $|L_\ell| = n_\ell$. Furthermore, we say that ℓ is a *dominant level* if $\sum_{i\in L_\ell} p_i > 1/2$; else, ℓ is a *non-dominant level*. Note that there is either no dominant level or a single unique dominant level.

Let X_ℓ be the random variable denoting the number of pages in L_ℓ that do not appear among the first k distinct pages in any particular sequence of page requests. Let M_ℓ be the multiset of page requests up to and including the first time all the pages in L_ℓ are requested. We denote $Y_\ell = |M_\ell|$ and Z_ℓ to be the number of page requests from M_ℓ to pages in $\mathbb{T}$.

We need certain structural properties of these random variables.

▶ **Lemma 12.** *The following three properties hold for* X_ℓ, Y_ℓ, *and* Z_ℓ*:*

(i) *For any level* ℓ *with* $n_\ell > 0$,

$$\sum_{i\in L_\ell} p_i q_i \le \frac{1}{2^\ell} \cdot \sum_{i\in L_\ell} q_i = \frac{1}{2^\ell} \cdot \mathbb{E}[X_\ell]. \tag{4.3}$$

(ii) *Let ℓ be a level with $n_\ell > 0$, and let H_{n_ℓ} denote the n_ℓth Harmonic number. Then,*

$$\mathbb{E}[Y_\ell] \le 2^{\ell+1} \cdot H_{n_\ell}. \tag{4.4}$$

(iii) *Suppose ℓ is a non-dominant level. Then, we have*

$$\Pr[Z_\ell \ge t] \le \begin{cases} 2 \cdot p(\mathbb{T}) \cdot 2^{\ell+1} \cdot \mathscr{H} & \text{for } t = 1, \quad (4.5) \\ \left(2e^2 \cdot p(\mathbb{T}) \cdot 2^{\ell+1} \cdot \mathscr{H}\right)^t & \text{for } t \ge 2. \quad (4.6) \end{cases}$$

Next, using these properties, we bound the total contribution of any single level L_ℓ to $\sum_i p_i q_i$ (i.e., to the LHS in (4.2)):

▷ Claim 13. For any level ℓ, we have $\sum_{i \in L_\ell} p_i q_i \le 4\mathscr{H} \cdot p(\mathbb{T})$.

Proof. By (4.3), $\sum_{i \in L_\ell} p_i q_i \le \frac{\mathbb{E}[X_\ell]}{2^\ell} \le \frac{\mathbb{E}[Z_\ell]}{2^\ell}$ since $X_\ell \le Z_\ell$ for any level ℓ.

So, it suffices to bound $\mathbb{E}[Z_\ell]$. Recall that M_ℓ denotes the sequence of requests until all pages in L_ℓ are requested. Color every request in M_ℓ red, blue, or green depending on whether it is for a page in $\mathbb{T}$, for a page in $\mathbb{H}$ but not in L_ℓ, and for a page in L_ℓ, respectively. Variable Z_ℓ counts the number of red requests in M_ℓ.

Let N denote the number of green requests in M_ℓ and let Γ_i, for $i \ge 1$, be the number of red requests between the $(i-1)$st and ith green request. Note that the random variables Γ_i are independent and identically distributed; let Γ denote the common distribution. Furthermore, N is independent of the sequence $\Gamma_1, \Gamma_2, \ldots$. Therefore, by Wald's identity we have

$$\mathbb{E}[Z_\ell] = \sum_i \mathbb{E}[\Gamma_i] = \mathbb{E}[\Gamma] \cdot \mathbb{E}[N]. \tag{4.7}$$

We bound the two terms in the RHS of (4.7) separately. For $\mathbb{E}[N]$, we use standard coupon collector bounds since the probabilities of pages in level ℓ are uniform up to a factor of 2: $\mathbb{E}[N] \le 2n_\ell H_{n_\ell} \le 2n_\ell \cdot \mathscr{H}$.

To bound $\mathbb{E}[\Gamma]$, note that Γ denotes the number of red requests before the first green request. Denote $p(\ell) := \sum_{i \in L_\ell} p_i$ as the probability of a green request and recall that $p(\mathbb{T})$ is the probability of a red request. Then, $\mathbb{E}[\Gamma] = \left(\frac{p(\ell)}{p(\ell)+p(\mathbb{T})}\right)^{-1} - 1 = \frac{p(\mathbb{T})}{p(\ell)} \le \frac{p(\mathbb{T})}{n_\ell \cdot 2^{-(\ell+1)}}$.

Therefore, by (4.7), $\mathbb{E}[Z_\ell] = \mathbb{E}[\Gamma] \cdot \mathbb{E}[N] < \frac{p(\mathbb{T})}{n_\ell \cdot 2^{-(\ell+1)}} \cdot 2n_\ell \cdot \mathscr{H} = 4\mathscr{H} \cdot p(\mathbb{T}) \cdot 2^\ell$, thus yielding the claim. ◀

The above bound is sufficiently tight for levels ℓ where the probabilities p_i are comparable to (or smaller than) the value of the tail probability $p(\mathbb{T})$. However, in higher levels with larger probabilities, the above bound is quite loose and we need a stronger guarantee.

Let r_1 be the largest index i such that $p_i \ge 4e^2 \cdot \mathscr{H} \cdot p(\mathbb{T})$ and let ℓ_1 be the level that contains page p_{r_1}. Roughly speaking, our tighter bound for levels $\ell < \ell_1$ geometrically shrinks as we go to levels with larger probabilities; this will later allow us to add these bounds across levels in a lossless manner.

▷ Claim 14. For any non-dominant level $\ell < \ell_1$, we have $\sum_{i \in L_\ell} p_i q_i \le \frac{80\mathscr{H}^2 \cdot p(\mathbb{T})}{2^{\ell_1 - \ell}}$.

Proof Sketch. Since $X_\ell \le Z_\ell$, we get from Lemma 12*(i)* that

$$\sum_{i \in L_\ell} p_i q_i \le \frac{1}{2^\ell} \cdot \mathbb{E}[X_\ell] \le \Pr[X_\ell = 1 \mid Z_\ell = 1] \cdot \Pr[Z_\ell \ge 1] + \sum_{t \ge 2} \Pr[Z_\ell \ge t], \tag{4.8}$$

where we split between $X_\ell = 1$ and $X_\ell \ge 2$, and replaced X_ℓ with Z_ℓ. We bound $\sum_{t \ge 2} \Pr[Z_\ell \ge t]$ and $\Pr[Z_\ell \ge 1]$ using Lemma 12 *(iii)*. The interesting part is to bound $\Pr[X_\ell = 1 \mid Z_\ell = 1]$.

Note that since $Z_\ell = 1$, there is only one request from $\mathbb{T}$ before all pages in level ℓ are requested. Therefore, a necessary event for $X_\ell = 1 \wedge Z_\ell = 1$ is that the last page requested in $\mathbb{H}$ must be from level ℓ; in particular, page r_1 (in level $\ell_1 > \ell$) is requested before some page in L_ℓ. Let $\mathcal{E}$ denote this event; we bound

$$\Pr[X_\ell = 1 \mid Z_\ell = 1] \le \Pr[\mathcal{E} \mid Z_\ell = 1] = \sum_{t \ge 1} \Pr[\mathcal{E} \mid Z_\ell = 1, Y_\ell = t] \cdot \Pr[Y_\ell = t]. \tag{4.9}$$

Consider any fixed number of requests, say t. The conditional probability of any page in the head $\mathbb{H}$ increases when we decrease the number of requests for pages in the tail $\mathbb{T}$ among these t requests. Therefore,

$$\Pr[\mathcal{E} \mid Z_\ell = 1, Y_\ell = t] \le \Pr[\mathcal{E} \mid Z_\ell = 0, Y_\ell = t] = \frac{1/2^{\ell_1}}{1 - p(\mathbb{T})} \cdot t \le \frac{t}{2^{\ell_1 - 1}}, \text{ since } p(\mathbb{T}) < 1/2.$$

Here, we used that given $Z_\ell = 0$, the probability that any request is for page r_1 is $\frac{1/2^{\ell_1}}{1-p(\mathbb{T})}$. The claim can now be established by putting together the bounds we have obtained in (4.8). ◁

4.2 Splitting the Head

We have established bounds on the contribution of individual levels to $\sum_i p_i q_i$. We are now ready to put these together to bound the total contribution of the head $\mathbb{H}$. To do this, we subdivide the head $\mathbb{H} = [k]$ into three intervals $\mathbb{H}_1 := [1, r_1]$, $\mathbb{H}_2 = [r_1 + 1, r_2]$, and $\mathbb{H}_3 := [r_2 + 1, k]$ as follows: we define r_1 to be the largest index i such that $p_i \ge 4e^2 \mathscr{H} \cdot p(\mathbb{T})$ (this is the same definition of r_1 that we used to define the level ℓ_1), and r_2 to be the largest index i such that $p_i \ge p(\mathbb{T})/2^{\mathscr{H}}$. We will separately bound the contribution of each of these intervals to $\sum_{i=1}^{k} p_i q_i$.

▶ **Lemma 15** (Bounding $\mathbb{H}_1$). *The contribution of $\mathbb{H}_1$ is bounded as $\sum_{i \in \mathbb{H}_1} p_i q_i \le 88 \mathscr{H}^2 \cdot p(\mathbb{T})$.*

Proof Sketch. This follows by using Claim 13 for a dominant level, if it exists, and level ℓ_1, the last level in $\mathbb{H}_1$, and using Claim 14 for the remaining levels. ◀

▶ **Lemma 16** (Bounding $\mathbb{H}_2$). *The contribution of $\mathbb{H}_2$ is bounded as $\sum_{i \in \mathbb{H}_2} p_i q_i \le 12 \mathscr{H}^2 \cdot p(\mathbb{T})$.*

Proof. By construction, the number of levels in $\mathbb{H}_2$ is at most $\log(4e^2 \mathscr{H} \cdot 2^{\mathscr{H}}) \le 6\mathscr{H}$. The claim now follows from Claim 13. ◀

▶ **Lemma 17** (Bounding $\mathbb{H}_3$). *The contribution of $\mathbb{H}_3$ is bounded as $\sum_{i \in \mathbb{H}_3} p_i q_i \le 4p(\mathbb{T})$.*

Proof Sketch. To bound the contribution of $\mathbb{H}_3$, we note that $q_i \le 1$ for all pages i and $\sum_{i \in L_\ell} p_i$ for any level ℓ is at most $n_\ell \cdot \frac{1}{2^\ell} \le 2^{\mathscr{H}} \cdot \frac{1}{2^\ell}$ by Lemma 5. The lemma now follows by adding over all levels in $\mathbb{H}_3$ and using $p_i \le p(\mathbb{T})/2^{\mathscr{H}}$ for any page $i \in \mathbb{H}_3$. ◀

We conclude with the proof of the main theorem.

Proof of Theorem 9. By adding up the bounds from Observation 11 and Lemmas 15–17, we get

$$\sum_{i \in [n]} p_i q_i \le (100\mathscr{H}^2 + 5) \cdot p(\mathbb{T}) = O(\mathscr{H}^2 p(\mathbb{T})).$$

By Lemma 8, this implies that LRU is $O(\mathscr{H}^3)$-competitive for stochastic caching. ◀

5 The LFU Algorithm

In this section, we analyze the Least Frequently Used (LFU) algorithm for stochastic caching. Recall that in LFU, we maintain a count for every page of the number of times that page was requested, and we keep in cache the k pages with the highest count. When a page outside of cache is requested, we evict the lowest count page in order to serve the request, then revert back to the k pages with highest count if necessary. Clearly, in the limit, the empirical count distribution will converge to the true distribution, and LFU will eventually behave identically to Top-k. However, the "burn-in" period until convergence could be arbitrarily long as a function of n; e.g., if $p_k - p_{k+1} = \epsilon$, then to even distinguish these two pages with probability δ requires $\Omega(\log(1/\delta)/\epsilon^2)$ samples (e.g., [14]).

Despite this, we show that LFU achieves an additive loss of $O(\mathscr{H} \cdot k)$ in addition to the $O(p(\mathbb{T}))$ cost paid by Top-k. Since k is incurred as an initialization cost for any algorithm, we show:

▶ **Theorem 18.** *The* LFU *algorithm has a competitive ratio $O(\mathscr{H})$ for the stochastic caching problem.*

The general proof idea is to show that every page in $\mathbb{H}$ (i.e., top k) might be requested while it is outside of cache many times, but that after $O(\mathscr{H})$ such cache misses, its count will forever be higher (from that point on, with good probability) than the count of any page with less than half its probability. If this is the case, we can charge any cache miss due to a request to $\mathbb{T}$ to $T \cdot p(\mathbb{T})$. Each of the k pages in $\mathbb{H}$ will only miss $O(\mathscr{H})$ times before the only pages displacing it from cache are ones with very similar probability, and hence distinguishing between these does not matter; the total contribution from these pages in $\mathbb{H}$ is $O(k\mathscr{H} + T \cdot p(\mathbb{T}))$. Below, we formalize this argument.

Proof. Define a parameter $\gamma = 16 \log(\min(k, n_{\max})) = O(\mathscr{H})$. For every $i \in \mathbb{H}$ and $j \in [n]$ such that $p_j \le p_i/2$, define $\mathcal{E}_{ij}$ to be the bad event that if the first time t that the empirical count of i is at least γ, there is at least one time $t' \ge t$ such that the empirical count of j exceeds the count of i. We show that $\mathcal{E}_{ij}$ is extremely unlikely for well separated p_i and p_j.

▷ Claim 19. For all $i \in \mathbb{H}$, $j \in [n]$ such that $p_j \le p_i/2^h$ for some $h \ge 1$, we have $\Pr[\mathcal{E}_{ij}] \le 2^{-h\gamma/8}$.

The proof uses standard facts and we leave it to Appendix D. The last claim only gets easier if instead of considering a single page j with $p_j \le p_i/2^h$, we consider a group of pages whose cumulative probability is at most $p_i/2^h$, and ask what are the odds that any of these pages pass i in count. (Since, even as we grouped these pages, their total count would still not exceed that of i with the same probability as before.)

▶ **Corollary 20.** *For all $i \in \mathbb{H}$ and any subset $S \subseteq [n]$ such that $\sum_{j \in S} p_j \le p_i/2^h$ for some $h \ge 1$, we have $\Pr[\bigcup_{j \in S} \mathcal{E}_{ij}] \le 2^{-h\gamma/8}$.*

For every $i \in \mathbb{H}$, let R_i be the number of requests to page i until the count of i never goes below that of any page j with $p_j \le p_i/2$. Using Claim 19 and Corollary 20, we can show that the expected number of requests until we achieve this count separation can be upper bounded.

▷ Claim 21. For every $i \in \mathbb{H} \setminus D$ for some set D such that $\sum_{i \in D} p_i = 2p(\mathbb{T})$, we have $\mathbb{E}[R_i] \le 2\gamma$.

Note that if we had set $\gamma = \Theta(\log n)$, we could guarantee by a union bound over all pairs of pages i and j that every page $i \in \mathbb{H}$ sees at most γ requests before no page j with $p_j \le p_i/2$ ever has count higher than i (with high probability). However, we can get away with smaller γ using a more careful analysis, which we again defer to Appendix D. We now show how to conclude the proof.

Let S_t be the subset of pages i in $\mathbb{H}$ that are both outside of cache at time t and that have been requested at least R_i times by time t. In other words, these are the pages $i \in \mathbb{H}$ outside of cache at time t such that no page j with $p_j \le p_i/2$ will have a count exceeding that of page i after time t. Define an arbitrary perfect matching M between the pages of S_t and a subset of the pages from $\mathbb{T}$ in cache at time t. If page $i \in S_t$ is matched to page $M(i)$, then $p_{M(i)} \ge p_i/2$, since the count of $M(i)$ must be higher than that of i, and $i \in S_t$.

Let C^t be the k pages with highest counts at time t. Then we can write the total cost of the LFU algorithm as

$$\begin{aligned}
\sum_t \sum_{i=1}^{n} p_i \Pr\left[i \notin C^t\right] &\le \sum_t \sum_{i \notin S_t} p_i \Pr\left[i \notin C^t\right] + \sum_t \sum_{i \in S_t} p_i \Pr\left[i \notin C^t\right] + \sum_t p(\mathbb{T}) \\
&\le 2T \cdot p(\mathbb{T}) + \sum_{i \in \mathbb{H}} \mathbb{E}[R_i] + \sum_t \sum_{i \in S_t} p_i \Pr\left[i \notin C^t\right] + T \cdot p(\mathbb{T}) && (5.10) \\
&\le 2\gamma \cdot k + 2 \cdot \sum_t \sum_{i \in S_t} p_{M(i)} + 3T \cdot p(\mathbb{T}) && (5.11) \\
&\le 2\gamma \cdot k + 5T \cdot p(\mathbb{T}). && (5.12)
\end{aligned}$$

Step (5.10) follows because the expected number of misses for each page $i \in \mathbb{H}$ until it belongs to S_t is $\mathbb{E}[R_i]$, step (5.11) follows by Claim 21 and since $p_i \le 2p_{M(i)}$, and step (5.12) follows since the matched pages in cache form a subset of $\mathbb{T}$, and hence their total probability is less than $p(\mathbb{T})$. ◀

6 Closing Remarks

In this work, we study the stochastic caching problem as a way to move beyond worst-case competitive ratio analysis for online algorithms. We give new evidence for the effectiveness of the popular LRU and LFU caching policies, by showing that for many distributions, they incur approximation factors significantly better than classical bounds predict. En route, we identify subset entropy as an important parameter of the distribution governing the achievable competitive ratio of stochastic caching; this information-theoretic quantity might be of independent interest.

There are several avenues for future work. The first obvious direction is to either improve the competitive ratio of LRU from $O(\mathscr{H}^3)$ to $O(\mathscr{H})$, or show that this is not possible. In Appendix F, we show that there are distributions for which the cost of LRU is $\Omega(\mathscr{H} \cdot p(\mathbb{T}))$, meaning that our techniques can at best prove an $O(\mathscr{H}^2)$ bound.

Another direction is to generalize the result to more complex stochastic models, e.g., the Markov paging model [23] (this would entail developing an analogous quantity to subset entropy). Going beyond caching, we often prove lower bounds for randomized online algorithms by exhibiting pathological input distributions. Parameterizing what makes these distributions difficult and, more generally, understanding and quantifying the performance of an online algorithm through a fine-grained measure such as subset entropy are promising directions of future research.

References

1 Algorithms with predictions (ALPS). https://algorithms-with-predictions.github.io/.

2 Alfred V. Aho, Peter J. Denning, and Jeffrey D. Ullman. Principles of optimal page replacement. *J. ACM*, 18(1):80–93, 1971. doi:10.1145/321623.321632.

3 Susanne Albers, Sanjeev Arora, and Sanjeev Khanna. Page replacement for general caching problems. In *SODA*, pages 31–40, 1999. URL: http://dl.acm.org/citation.cfm?id=314500.314528.

4 Nikhil Bansal, Niv Buchbinder, and Joseph Naor. A primal-dual randomized algorithm for weighted paging. *J. ACM*, 59(4):19, 2012.

5 Nikhil Bansal, Niv Buchbinder, and Joseph Naor. Randomized competitive algorithms for generalized caching. *SICOMP*, 41(2):391–414, 2012. doi:10.1137/090779000.

6 Nikhil Bansal, Christian Coester, Ravi Kumar, Manish Purohit, and Erik Vee. Learning-augmented weighted paging. In *SODA*, pages 67–89, 2022. doi:10.1137/1.9781611977073.4.

7 Luca Becchetti. Modeling locality: A probabilistic analysis of LRU and FWF. In *ESA*, pages 98–109, 2004. doi:10.1007/978-3-540-30140-0_11.

8 Laszlo A. Belady. A study of replacement algorithms for a virtual-storage computer. *IBM Systems Journal*, 5(2):78–101, 1966. doi:10.1147/SJ.52.0078.

9 Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge U. Press, 1998.

10 Allan Borodin, Sandy Irani, Prabhakar Raghavan, and Baruch Schieber. Competitive paging with locality of reference. *JCSS*, 50(2):244–258, 1995. doi:10.1006/JCSS.1995.1021.

11 Sébastien Bubeck, Michael B Cohen, Yin Tat Lee, James R Lee, and Aleksander Madry. K-server via multiscale entropic regularization. In *STOC*, pages 3–16, 2018. doi:10.1145/3188745.3188798.

12 Niv Buchbinder, Shahar Chen, and Joseph Naor. Competitive analysis via regularization. In *SODA*, pages 436–444, 2014. doi:10.1137/1.9781611973402.32.

13 Niv Buchbinder, Joseph Seffi Naor, et al. The design of competitive online algorithms via a primal–dual approach. *Foundations and Trends® in Theoretical Computer Science*, 3(2–3):93–263, 2009. doi:10.1561/0400000024.

14 Clément L Canonne. A short note on an inequality between KL and TV. *arXiv*, 2202.07198, 2022. arXiv:2202.07198.

15 Nikhil R. Devanur, Kamal Jain, and Robert D. Kleinberg. Randomized primal-dual analysis of RANKING for online bipartite matching. In *SODA*, pages 101–107, 2013. doi:10.1137/1.9781611973105.7.

16 Guy Even, Moti Medina, and Dror Rawitz. Online generalized caching with varying weights and costs. In *SPAA*, pages 205–212, 2018. doi:10.1145/3210377.3210404.

17 Amos Fiat, Richard M. Karp, Michael Luby, Lyle A. McGeoch, Daniel Dominic Sleator, and Neal E. Young. Competitive paging algorithms. *J. ALG*, 12(4):685–699, 1991. doi:10.1016/0196-6774(91)90041-V.

18 Peter A. Franaszek and Terry J. Wagner. Some distribution-free aspects of paging algorithm performance. *J. ACM*, 21(1):31–39, 1974. doi:10.1145/321796.321800.

19 Anupam Gupta, Ravishankar Krishnaswamy, Amit Kumar, and Debmalya Panigrahi. Elastic caching. In *SODA*, pages 143–156, 2019. doi:10.1137/1.9781611975482.10.

20 Anupam Gupta, Amit Kumar, and Debmalya Panigrahi. Caching with time windows and delays. *SICOMP*, 51(4):975–1017, 2022. doi:10.1137/20M1346286.

21 S. Irani. Randomized weighted caching with two page weights. *Algorithmica*, 32(4):624–640, 2002. doi:10.1007/S00453-001-0095-6.

22 Sandy Irani. Page replacement with multi-size pages and applications to web caching. In *STOC*, pages 701–710, 1997. doi:10.1145/258533.258666.

23 Anna R. Karlin, Steven J. Phillips, and Prabhakar Raghavan. Markov paging. *SICOMP*, 30(3):906–922, 2000. doi:10.1137/S0097539794268042.

24 Elias Koutsoupias and Christos H. Papadimitriou. Beyond competitive analysis. *SICOMP*, 30(1):300–317, 2000. doi:10.1137/S0097539796299540.

25 Ravi Kumar, Manish Purohit, Zoya Svitkina, and Erik Vee. Interleaved caching with access graphs. In *SODA*, pages 1846–1858, 2020. doi:10.1137/1.9781611975994.113.

26 Carsten Lund, Steven J. Phillips, and Nick Reingold. Paging against a distribution and IP networking. *JCSS*, 58(1):222–232, 1999. doi:10.1006/JCSS.1997.1498.

27 Thodoris Lykouris and Sergei Vassilvitskii. Competitive caching with machine learned advice. In *ICML*, pages 3302–3311, 2018. URL: http://proceedings.mlr.press/v80/lykouris18a.html.

28 Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge U. Press, 1995. doi:10.1017/CBO9780511814075.

29 Chirag Pabbaraju and Ali Vakilian. New and improved bounds for Markov paging. In *ICALP*, pages 123:1–123:20, 2025. doi:10.4230/LIPIcs.ICALP.2025.123.

30 Konstantinos Panagiotou and Alexander Souza. On adequate performance measures for paging. In *STOC*, pages 487–496, 2006. doi:10.1145/1132516.1132587.

31 Gopal Pandurangan and Eli Upfal. Entropy-based bounds for online algorithms. *TALG*, 3(1):7:1–7:19, 2007. doi:10.1145/1219944.1219953.

32 Tim Roughgarden, editor. *Beyond the Worst-Case Analysis of Algorithms*. Cambridge U. Press, 2020.

33 Daniel D. Sleator and Robert E. Tarjan. Amortized efficiency of list update and paging rules. *C. ACM*, 28(2):202–208, 1985. doi:10.1145/2786.2793.

34 J Michael Steele. *Stochastic Calculus and Financial Applications. Vol I*. Springer, 2001.

35 Andrew Chi-Chih Yao. Probabilistic computations: Toward a unified measure of complexity (extended abstract). In *FOCS*, pages 222–227, 1977. doi:10.1109/SFCS.1977.24.

36 Neal E. Young. On-line paging against adversarially biased random inputs. *J. ALG*, 37(1):218–235, 2000. doi:10.1006/JAGM.2000.1099.

37 Neal E. Young. Online paging and caching. In *Encyclopedia of Algorithms*, pages 1457–1461. Springer, 2016. doi:10.1007/978-1-4939-2864-4_267.

A Missing Proofs from Section 2

Proof of Lemma 5. Note that for every ℓ, the distribution $\mathcal{D}_{S_\ell}$ is uniform up to a factor of 2 (i.e., the maximum probability is at most twice the minimum probability). As noted, the Shannon entropy of such a distribution is $\Theta(\log n_\ell)$, where n_ℓ is the size of its support. We choose S as follows: if $n_{\max} \le k$, then $S = \arg\max_\ell |S_\ell|$ and hence $|S| = n_{\max}$; else, S is any k-sized subset of $\arg\max_\ell |S_\ell|$ and hence $|S| = k$. Then, we get

$$\mathscr{H}(\mathcal{D}, k) \ge \mathrm{H}(\mathcal{D}_S) = \Omega(\log |S|) = \Omega(\log(\min\{k, n_{\max}\})).$$

We now show the other direction, by proving that for any distribution $\mathcal{D}'$ with $\max_\ell |\{u \in U : 1/2^{\ell+1} < \mathcal{D}'(u) \le 1/2^\ell)\}| \le N$, we have that $H(\mathcal{D}') \le O(\log N))$. Note that $N \le \min\{2n_{\max}, k\}$ for $\mathcal{D}_S$, where $S \subseteq [n]$ is any subset of cardinality at most k, since pages in S whose probabilities in $\mathcal{D}_S$ differ by at most a factor of two can come from at most two adjacent levels of the original distribution $\mathcal{D}$.

Letting $S'_\ell := \{u : 1/2^{\ell+1} < \mathcal{D}'(u) \le 1/2^\ell\}$, we have,

$$\mathrm{H}(\mathcal{D}') = -\sum_u \mathcal{D}'(u) \cdot \log \mathcal{D}'(u) = -\sum_\ell \sum_{u \in S'_\ell} \mathcal{D}'(u) \cdot \log \mathcal{D}'(u).$$

Now, for any $u \in S'_\ell$, we have $\mathcal{D}'(u) \le 1/2^\ell$ and $-\log \mathcal{D}'(u) = \log 1/\mathcal{D}'(u) < \log 2^{\ell+1} = \ell + 1$.

Therefore, $\mathrm{H}(\mathcal{D}') \le \sum_\ell n_\ell \cdot \frac{\ell+1}{2^\ell}$. where $n_\ell := |S'_\ell|$.

To bound the RHS of this inequality, we use a factor-revealing LP. We note that $\mathrm{H}(\mathcal{D}')$ is at most the optimal primal objective in the following LP:

$$\text{maximize } \sum_{\ell \geq 1} n_\ell \cdot \frac{\ell+1}{2^\ell} \text{ subject to } \sum_{\ell} \frac{n_\ell}{2^\ell} \leq 1 \text{ and } \{n_\ell \leq N\}_{\forall \ell}.$$

The dual LP is

$$\text{minimize } \alpha + N \sum_{\ell} \beta_\ell, \text{ subject to } \left\{ \frac{\alpha}{2^\ell} + \beta_\ell \geq \frac{\ell+1}{2^\ell} \right\}_{\forall \ell}.$$

Now, a feasible dual solution is given by:

$$\alpha = \log N \qquad \beta_\ell = \begin{cases} \frac{\ell+1}{2^\ell} & \text{if } \ell > \log N - 1 \\ 0 & \text{if } \ell \leq \log N - 1. \end{cases}$$

The dual objective for this solution is $O(\log N) = O(\log(\min\{k, n_{\max}\}))$. Therefore, $\mathscr{H}(\mathcal{D}, k) = \mathrm{H}(\mathcal{D}_S) \leq O(\log(\min\{k, n_{\max}\})) = O(\log(\min\{k, \max_\ell |S_\ell|\}))$. ◀

B Missing Proofs from Section 3

We first define our true dual vector y. For any time t, let $G(t) \subseteq [n]$ be the set of pages $i \notin [k]$, for which the interval $I(i, \rho(i,t))$ has $|I(i, \rho(i,t))| \leq 3/p_i$. Let $\sigma^t_1, \ldots, \sigma^t_{m^t}$ be the pages of $G(t)$ in decreasing order of p_i. Then define $S^t_j = [k] \cup \{\sigma^t_1, \ldots, \sigma^t_j\}$, and

$$y^t_S = \begin{cases} p_{\sigma^t_j} - p_{\sigma^t_{j+1}} & \text{if } S = S^t_j \text{ and } j < m^t. \\ p_{\sigma^t_{m^t}} & \text{if } S = S^t_j \text{ and } j = m^t. \\ 0 & \text{otherwise.} \end{cases}$$

In words, we define dual variables only for prefixes of the pages in decreasing order of probability (after removing pages whose intervals exceed their expected length by a constant), and include the k pages in $\mathbb{H}$ in all these prefixes.

We argue that this dual is approximately feasible with constant probability, and that its objective value is sufficiently large. We start with the first claim. Let $n_{\max} := \max_\ell |S_\ell|$ in Lemma 5 and let $\eta := \min\{n_{\max}, k\}$. Recall that by Lemma 5, we have that $\mathscr{H} = \Theta(\log \eta)$.

▶ **Lemma 22.** *Fix a time step t. With probability at least $3/4$, for all $i \in [n]$, the solution y satisfies the dual constraints associated with $(i, \rho(i,t))$ up to a factor of $3\ln(2\eta)$.*

Proof. Since t is fixed, for convenience we define $I(i) := I(i, \rho(i,t))$.

For pages $i \in [n] \setminus ([k] \cup G(t))$, for all $t' \in I(i)$ and S such that $i \in S$, note that $y^{t'}_S = 0$. So, the corresponding dual constraint is satisfied vacuously.

Next, consider constraints associated with pages $i \in G(t)$. For all $t' \in I(i)$, if $j^*(t')$ is the index of i in $\sigma^{t'}_1, \ldots, \sigma^{t'}_{m^{t'}}$, then by design

$$\sum_{S: i \in S} y^{t'}_S = \sum_{j=j^*(t)}^{m^{t'}-1} \left(p_{\sigma^{t'}_j} - p_{\sigma^{t'}_{j+1}} \right) + p_{\sigma^{t'}_{m^{t'}}} = p_{\sigma^{t'}_{j^*(t')}} = p_i,$$

which means

$$\sum_{t' \in I(i)} \sum_{S: i \in S} y^{t'}_S = p_i \cdot |I(i)| \leq p_i \cdot \frac{3}{p_i} = 3.$$

It remains to handle constraints associated with $i \in [k]$. This time, for all $t' \in I(i)$, we have

$$\sum_{S: i \in S} y^{t'}_S = \sum_{j=1}^{m^{t'}-1} \left(p_{\sigma^{t'}_j} - p_{\sigma^{t'}_{j+1}} \right) + p_{\sigma^{t'}_{m^{t'}}} = p_{\sigma^{t'}_1} \leq p_k,$$

and hence

$$\Pr\left[\sum_{t'\in I(i)}\sum_{S:\,i\in S} y_S^{t'} \geq 3\ln(2\eta)\right] \leq \Pr\left[|I(i)|\cdot p_k \geq 3\ln(2\eta)\right] \leq \Pr\left[|I(i)| \geq E[|I(i)|]\cdot 3\ln(2\eta)\cdot\frac{p_i}{p_k}\right].$$

Since $|I(i,j)|$ is a geometric random variable with parameter p_i, we get that

$$\Pr\left[\sum_{t'\in I(i)}\sum_{S:\,i\in S} y_S^{t'} \geq 3\ln(2\eta)\right] \leq (1-p_i)^{E[|I(i)|]\cdot 3\ln(2\eta)\cdot\frac{p_i}{p_k}} \leq e^{-3\ln(2\eta)\cdot\frac{p_i}{p_k}} \leq (2\eta)^{-3\cdot\frac{p_i}{p_k}}.$$

Let L_ℓ be the pages i in $[k]$ such that $p_k\cdot 2^{\ell+1} > p_i \geq p_k\cdot 2^\ell$. Note that $|L_\ell| \leq 2n_{\max}$ since it contains pages from at most two sets $S_{\ell'}, S_{\ell'+1}$ for some ℓ'. Moreover, $|L_\ell| \leq k$. Hence, $|L_\ell| \leq 2\eta$.

Now, let $\mathcal{E}$ be the event that there exists an $i \in [k]$ for which $\sum_{t'\in I(i)}\sum_{S:\,i\in S} y_S^{t'} \geq 3\ln(2\eta)$, i.e., the dual constraint for page i is violated beyond a factor of $3\ln(2\eta)$. By a union bound,

$$\Pr[\mathcal{E}] \leq \sum_{\ell\geq 0}\sum_{\substack{i\in[k]:\\ i\in L_\ell}} (2\eta)^{-3\cdot\frac{p_i}{p_k}} \leq \sum_{\ell\geq 0}\sum_{\substack{i\in[k]:\\ i\in L_\ell}} (2\eta)^{-3\cdot 2^\ell} \leq \sum_{\ell\geq 0} 2\eta\cdot(2\eta)^{-3\cdot 2^\ell} \leq (2\eta)^{-2} \leq \frac{1}{4}. \qquad \blacktriangleleft$$

▶ **Lemma 23.** *The expected objective of the dual y at time t is at least $2p(\mathbb{T})/3$.*

Proof. By construction, the dual objective at time t is

$$\sum_S (|S|-k)\cdot y_S^t = m^t\cdot p_{\sigma^t_{m^t}} + \sum_{j=1}^{m^t-1} j\cdot(p_{\sigma^t_j} - p_{\sigma^t_{j+1}}) = \sum_{i\in G(t)} p_i.$$

By a Markov bound, every page $i \in [n]\setminus[k]$ appears in $G(t)$ with probability at least $2/3$, hence the claim. $\blacktriangleleft$

Using these, we can conclude the proof of Lemma 8, which in turn implies Theorem 6 by weak duality.

Proof of Lemma 8. From Lemma 22 and Lemma 23, there exists a dual solution y such that:

(a) At every time t, with probability $\geq 3/4$, solution y simultaneously satisfies all constraints (i,j) such that $t \in I(i,j)$ up to a multiplicative factor of $3\ln(2\eta)$.

(b) The expected dual objective value of this solution y is $2p(\mathbb{T})/3$.

Define a modified dual solution $\widehat{y}$ as follows. For every time t, if there exists a dual constraint (i,j) such that $t \in I(i,j)$ that is violated by a multiplicative factor greater than $3\ln(2\eta)$ (call this event $\mathcal{E}(t)$), then set $\widehat{y}_S^t = 0$ for all $S \subseteq [n]$. Otherwise, set $\widehat{y}_S^t = y_S^t/(3\ln(2\eta))$.

The new dual is always feasible. Furthermore, when $y_S^t > 0$, we have $\widehat{y}_S^t = y_S^t/(3\ln(2\eta))$ with probability at least $3/4$. Hence, by the linearity of expectation, the dual objective of $\widehat{y}$ is

$$\begin{aligned}\sum_S (|S|-k)\cdot\mathbb{E}\big[\widehat{y}_S^t\big] &\geq \frac{1}{3\ln(2\eta)}\sum_S (|S|-k)\cdot\left(\mathbb{E}\big[y_S^t\big] - \mathbb{E}\big[y_S^t \,\big|\, \mathcal{E}(t)\big]\cdot\Pr[\mathcal{E}(t)]\right)\\ &\geq \frac{2p(\mathbb{T})/3 - p(\mathbb{T})/4}{3\ln(2\eta)} \qquad \text{(B.13)}\\ &\geq \frac{5p(\mathbb{T})/12}{3\ln(2\cdot\min\{k,n_{\max}\})},\end{aligned}$$

where in (B.13) we used $\sum_S (|S|-k)\cdot y_S^t \leq p(\mathbb{T})$ with probability 1, together with Lemmas 22 and 23.

The characterization of $\mathscr{H} = \Theta(\log\min\{k, n_{\max}\})$ from Lemma 5 concludes the proof. $\blacktriangleleft$

C Missing Proofs from Section 4

C.1 Proof of Lemma 12

Proof of Lemma 12. We prove the three parts separately.

(i) It follows from the fact that $0 \le X_\ell \le n_\ell$, and the fact that $\mathbb{E}[X_\ell] = \sum_{i \in L_\ell} q_i$.

(ii) This fact follows from standard coupon collector bounds, since each request is for a page in L_ℓ with probability $\ge 1/2^{\ell+1}$ and there are n_ℓ distinct pages in L_ℓ.

(iii) Since ℓ is a non-dominant level, even after conditioning on the page requests being from level ℓ, the probability that any request is for a page in $\mathbb{T}$ is at most $2p(\mathbb{T})$. We use this fact in the rest of this proof.

For $t = 1$, we have

$$\begin{aligned}\Pr[Z_\ell \ge 1] \le \mathbb{E}[Z_\ell] &= \sum_{L \ge 1} \mathbb{E}[Z_\ell \mid Y_\ell = L] \cdot \Pr[Y_\ell = L] \le 2p(\mathbb{T}) \sum_{L \ge 1} L \cdot \Pr[Y_\ell = L] \\ &= 2p(\mathbb{T}) \cdot \mathbb{E}[L] \le p(\mathbb{T}) \cdot 2^{\ell+1} \cdot \mathscr{H}.\end{aligned}$$

For $t \ge 2$, let $\mu := \mathbb{E}[Y_\ell] \le 2^{\ell+1} \cdot \mathscr{H}$. Now, the value of Y_ℓ can be partitioned into intervals $(e(\gamma-1)\mu, e\gamma\mu]$ for integers $\gamma \ge 1$. Using this partitioning, we get

$$\begin{aligned}\Pr[Z_\ell \ge t] &= \sum_{\gamma \ge 1} \Pr[Z_\ell \ge t \mid e(\gamma-1)\mu < Y_\ell \le e\gamma\mu] \cdot \Pr[e(\gamma-1)\mu < Y_\ell \le e\gamma\mu] \\ &= \sum_{\gamma \ge 1} \Pr[Z_\ell \ge t \mid Y_\ell \le e\gamma\mu] \cdot \Pr[Y_\ell > e(\gamma-1)\mu].\end{aligned}$$

Using the Binomial distribution, we get $\Pr[Z_\ell \ge t \mid Y_\ell \le e\gamma\mu] \le \binom{e\gamma\mu}{t}(2p(\mathbb{T}))^t \le \left(\frac{e^2\gamma\mu}{t}\right)^t (2p(\mathbb{T}))^t$, where the last step used Stirling's approximation.

We now bound the second term, $\Pr[Y_\ell > e(\gamma-1)\mu]$. Observe that for $Y_\ell > e(\gamma-1)\mu$ to hold, each of the first $\gamma - 1$ blocks of $e\mu$ requests must fail to request all the pages from level ℓ. By Markov's inequality, since $\mu = \mathbb{E}[Y_\ell]$, each block of $e\mu$ requests fails to request all pages from level ℓ with probability at most $1/e$, and since these failure events are independent, $\Pr[Y_\ell > e(\gamma-1)\mu] \le e^{-(\gamma-1)}$. Therefore, for $\gamma \ge 1$, we have $\Pr[Y_\ell > e(\gamma-1)\mu] \le e^{-(\gamma-1)}$. Putting these together, we get

$$\begin{aligned}\Pr[Z_\ell \ge t] &\le \sum_{\gamma \ge 1} \left(\frac{e^2\gamma\mu}{t}\right)^t (2p(\mathbb{T}))^t \cdot e^{-(\gamma-1)} \\ &\le \left(\frac{e^2\mu}{t}\right)^t \cdot (2p(\mathbb{T}))^t \cdot \sum_{\gamma=1}^{\infty} \frac{\gamma^t}{e^{\gamma-1}} \le \left(\frac{e^2\mu}{t}\right)^t \cdot (2p(\mathbb{T}))^t \cdot \left(\sum_{\gamma=1}^{2t\ln(t)-1} \frac{\gamma^t}{e^{\gamma-1}} + \sum_{\gamma=2t\ln t}^{\infty} \frac{\gamma^t}{e^{\gamma-1}}\right) \\ &\le \left(\frac{e^2\mu}{t}\right)^t \cdot (2p(\mathbb{T}))^t \cdot \left((2t\ln(t)-1) \cdot \frac{t^t}{e^{t-1}} + 1\right) \qquad \text{(C.14)} \\ &\le \left(2e^2\mu\right)^t \cdot p(\mathbb{T})^t. \qquad \text{(since } t \ge 2\text{)}\end{aligned}$$

Step (C.14) follows since the expression $\gamma^t/e^{\gamma-1}$ is maximized at $\gamma = t$, so the first partial sum can be bounded by its length times the maximum t^t/e^{t-1}. The suffix sum converges to less than one. ◀

C.2 Proof of Claim 14

Proof of Claim 14. Recall that by (4.3), we have that for any level ℓ, $\sum_{i \in L_\ell} p_i q_i \le \frac{1}{2^\ell} \cdot \mathbb{E}[X_\ell]$, where X_ℓ denotes the number of pages in L_ℓ that do not appear among the first k distinct pages. We proceed to bound $\mathbb{E}[X_\ell]$. By definition, $X_\ell \le Z_\ell$. Therefore, we can write

$$\begin{aligned}\mathbb{E}[X_\ell] &= \sum_{t\geq 1} t\cdot \Pr[X_\ell = t] = \Pr[X_\ell = 1] + \sum_{t\geq 2} t\cdot \Pr[X_\ell = t] = \Pr[X_\ell = 1] + \sum_{t\geq 2}\Pr[X_\ell \geq t]\\ &\leq \Pr[X_\ell = 1 \mid Z_\ell = 1]\cdot \Pr[Z_\ell = 1] + \sum_{t\geq 2}\Pr[Z_\ell \geq t]\\ &\leq \Pr[X_\ell = 1 \mid Z_\ell = 1]\cdot \Pr[Z_\ell \geq 1] + \sum_{t\geq 2}\Pr[Z_\ell \geq t]. \end{aligned} \tag{C.15}$$

We can bound factors of (C.15) of the form $\Pr[Z_\ell \geq t]$ using (4.6). We get

$$\sum_{t\geq 2}\Pr[Z_\ell \geq t] \leq \sum_{t\geq 2}\left(2e^2\cdot p(\mathbb{T})\cdot 2^{\ell+1}\cdot \mathscr{H}\right)^t \leq 2\left(2e^2\cdot p(\mathbb{T})\cdot 2^{\ell+1}\cdot \mathscr{H}\right)^2,$$

where in the last step, we used

$$2e^2\cdot p(\mathbb{T})\cdot 2^{\ell+1}\cdot \mathscr{H} \leq 2e^2\cdot p(\mathbb{T})\cdot 2^{\ell_1}\cdot \mathscr{H} \leq \frac{2e^2\cdot p(\mathbb{T})\cdot \mathscr{H}}{4e^2\mathscr{H}\cdot p(\mathbb{T})} = 1/2.$$

Thus, $\sum_{t\geq 2}\Pr[Z_\ell \geq t] \leq 2\left(2e^2\cdot p(\mathbb{T})\cdot 2^{\ell+1}\cdot \mathscr{H}\right)^2 \leq \frac{\mathscr{H}\cdot p(\mathbb{T})}{2^{\ell_1-2\ell-6}}$, since $p(\mathbb{T}) \leq 1/(4e^2\mathscr{H}\cdot 2^{\ell_1})$.

It remains to bound the first part of (C.15). From (4.5), we have that $\Pr[Z_\ell \geq 1] \leq 2p(\mathbb{T})\cdot 2^{\ell+1}\cdot \mathscr{H}$. We are left to bound $\Pr[X_\ell = 1 \mid Z_\ell = 1]$. Since there is only one request from $\mathbb{T}$ before all pages in level ℓ are requested, a necessary event for X_ℓ to be 1 is that the last page requested in $\mathbb{H}$ must be from level ℓ. In particular, consider page r_1 which is in level ℓ_1 and therefore not in level $\ell < \ell_1$. Let $\mathcal{E}$ denote the event that page r_1 is requested before all pages in level ℓ are requested. Then,

$$\Pr[X_\ell = 1 \mid Z_\ell = 1] \leq \Pr[\mathcal{E} \mid Z_\ell = 1] = \sum_{t\geq 1}\Pr[\mathcal{E} \mid Z_\ell = 1, Y_\ell = t]\cdot \Pr[Y_\ell = t]. \tag{C.16}$$

Consider any fixed number of requests, say t. The conditional probability of any page in the head $\mathbb{H}$ increases when we decrease the number of requests for pages in the tail $\mathbb{T}$ among these t requests. Therefore, $\Pr[\mathcal{E} \mid Z_\ell = 1, Y_\ell = t] \leq \Pr[\mathcal{E} \mid Z_\ell = 0, Y_\ell = t]$.

Now, let $I_t(\mathcal{E})$ denote the number of requests for page r_1 among the first t requests. Given that $Z_\ell = 0$, the probability that any request is for page r_1 is $2^{-\ell_1}/(1-p(\mathbb{T}))$. Therefore,

$$\Pr[\mathcal{E} \mid Z_\ell = 1, Y_\ell = t] \leq \mathbb{E}[I_t(\mathcal{E}) \mid Z_\ell = 0] = \frac{1/2^{\ell_1}}{1-p(\mathbb{T})}\cdot t \leq \frac{t}{2^{\ell_1-1}},$$

since $p(\mathbb{T}) < 1/2$. Substituting in (C.16), we get

$$\begin{aligned}\Pr[X_\ell = 1 \mid Z_\ell = 1] &= \sum_{t\geq 1}\Pr[\mathcal{E} \mid Z_\ell = 1, Y_\ell = t]\cdot \Pr[Y_\ell = t] \leq \sum_{t\geq 1}\frac{t}{2^{\ell_1-1}}\cdot \Pr[Y_\ell = t]\\ &\leq \frac{\mathbb{E}[Y_\ell]}{2^{\ell_1-1}} \leq \frac{\mathscr{H}\cdot 2^{\ell+1}}{2^{\ell_1-1}} = \frac{4\mathscr{H}}{2^{\ell_1-\ell}},\end{aligned}$$

where we again used (4.4) in the last inequality. Therefore,

$$\Pr[X_\ell = 1 \mid Z_\ell = 1]\cdot \Pr[Z_\ell \geq 1] \leq \frac{4\mathscr{H}}{2^{\ell_1-\ell}}\cdot 2p(\mathbb{T})\cdot 2^{\ell+1}\cdot \mathscr{H} = \frac{\mathscr{H}^2\cdot p(\mathbb{T})}{2^{\ell_1-2\ell-4}}.$$

This bounds the first expression in (C.15).

Substituting our bounds for both expressions in (C.15), we get

$$\mathbb{E}[X_\ell] \leq \frac{\mathscr{H}^2\cdot p(\mathbb{T})}{2^{\ell_1-2\ell-4}} + \frac{\mathscr{H}\cdot p(\mathbb{T})}{2^{\ell_1-2\ell-6}} \leq \frac{80\mathscr{H}^2\cdot p(\mathbb{T})}{2^{\ell_1-2\ell}}.$$

Therefore,

$$\sum_{i\in L_\ell} p_i q_i \leq \frac{1}{2^\ell}\cdot \mathbb{E}[X_\ell] \leq \frac{1}{2^\ell}\cdot \frac{80\mathscr{H}^2\cdot p(\mathbb{T})}{2^{\ell_1-2\ell}} = \frac{80\mathscr{H}^2\cdot p(\mathbb{T})}{2^{\ell_1-\ell}}. \qquad \triangleleft$$

C.3 Proof of Lemma 15

Proof of Lemma 15. We use Claim 13 to bound both a dominant level, if it exists, and the last level in $\mathbb{H}_1$, i.e., ℓ_1. For both these levels $\ell = \ell^*$ and $\ell = \ell_1$, we get $\sum_{i \in L_\ell} p_i q_i \le 4\mathscr{H} \cdot p(\mathbb{T})$, by Claim 13.

For non-dominant levels $\ell < \ell_1$, we use Claim 14. Thus, $\sum_{\ell<\ell_1} \sum_{i \in L_\ell} p_i q_i \le \sum_{\ell_1 - \ell \ge 1} \frac{80\mathscr{H}^2 \cdot p(\mathbb{T})}{2^{\ell_1 - \ell}} \le 80\mathscr{H}^2 \cdot p(\mathbb{T})$. Summing the bounds, we get $\sum_{i \in \mathbb{H}_1} p_i q_i \le 4\mathscr{H} \cdot p(\mathbb{T}) + 80\mathscr{H}^2 \cdot p(\mathbb{T}) \le 88\mathscr{H}^2 \cdot p(\mathbb{T})$. ◀

C.4 Proof of Lemma 17

Proof of Lemma 17. Let ℓ_3 be the first level ℓ that contains pages from $\mathbb{H}_3$, i.e., all pages in $\mathbb{H}_3$ are in levels ℓ_3 and above. Then, $\sum_{i \in \mathbb{H}_3} p_i q_i \le \sum_{\ell \ge \ell_3} \frac{1}{2^\ell} \sum_{i \in L_\ell} q_i \le \sum_{\ell \ge \ell_3} \frac{1}{2^\ell} \cdot n_\ell$, since $q_i \le 1$ by virtue of being a probability.

By Lemma 5, for any level ℓ, the number pages n_ℓ is at most $2^{\mathscr{H}}$. Therefore,

$$\sum_{i \in \mathbb{H}_3} p_i q_i \le \sum_{\ell \ge \ell_3} \frac{1}{2^\ell} \cdot n_\ell \le 2^{\mathscr{H}} \cdot \sum_{\ell \ge \ell_3} \frac{1}{2^\ell} \le 2^{\mathscr{H}} \cdot \frac{1}{2^{\ell_3 - 1}}.$$

Note that $p_i \le p(\mathbb{T})/2^{\mathscr{H}}$ for any page $i \in \mathbb{H}_3$. If page i is in level ℓ, then $p_i \ge 1/2^{\ell+1}$. Combining the two, we get $\frac{1}{2^{\ell+1}} \le p_i \le \frac{p(\mathbb{T})}{2^{\mathscr{H}}}$, and, in particular, $\frac{1}{2^{\ell_3 - 1}} \le \frac{4p(\mathbb{T})}{2^{\mathscr{H}}}$. Putting these together, we get $\sum_{i \in \mathbb{H}_3} p_i q_i \le 2^{\mathscr{H}} \cdot \frac{1}{2^{\ell_3 - 1}} \le 4p(\mathbb{T})$. ◀

D Missing Proofs from Section 5

D.1 Proof of Claim 19

Proof of Claim 19. Fix $i \in \mathbb{H}$ and j such that $p_j \le p_i/2$. Consider the following random walk on the line that starts at 0, moves +1 with probability $p_i/(p_i + p_j) \ge 1 - 1/(2^h + 1)$, and moves −1 with probability $p_j/(p_i + p_j) \le 1/(2^h + 1)$. After γ time steps, the position is at least $\gamma/2$ with probability at least $1 - 2^{-h\gamma/4}$. Indeed, observe that the probability the walk is at position $< \gamma/2$ after γ time steps is the probability that γ i.i.d. Bernoullis $X_1, \ldots, X_\gamma$ with probability $q = 1/(2^h + 1)$ sum to $\gamma/4$. The sum $X = \sum_i X_i$ has mean $\mu = q\gamma$ and if we set $\delta = \gamma/(4\mu) - 1$, by a Chernoff bound,

$$\Pr\left[\sum_i X_i \ge \frac{\gamma}{4}\right] = \Pr\left[\sum_i X_i \ge (1+\delta)\mu\right] \le \frac{e^{(1+\delta)\mu}}{(1+\delta)^{(1+\delta)\mu}} \le e^{\gamma/4}(4q)^{-\gamma/4} \le 2^{-h\gamma/4}.$$

A random walk with bias $1 - 1/(2^h + 1)$ at position z returns to 0 with probability exactly 2^{-hz} (see, e.g., [34, page 6]).

Now, it is easy to see that the event $\mathcal{E}_{ij}$ has the same probability as that of this random walk returning to 0 after γ time steps. Therefore, if $\mathcal{E}_{ij}$ holds either (1) the random walk was at position $\le \gamma/4$ after γ steps, or (2) starting at position $\gamma/4$, the walk returned to 0 after a finite amount of time. By a union bound, $\Pr[\mathcal{E}_{ij}] \le 2^{-h\gamma/8}$. ◁

D.2 Proof of Claim 21

Proof of Claim 21. Recall that $\gamma = 16\mathscr{H} = 16 \log \min(k, n_{\max})$. We do a case analysis.

Case 1. Suppose that $k \le n_{\max}$, so $\gamma = 16 \log k$. Recalling that the pages of $\mathbb{T}$ are ordered by decreasing probability, define $G_\ell := \{i \in \mathbb{T} \mid i \ge 2k+1,\ i = \ell \mod k\}$. Define $q := 2 \cdot \sum_{i \in G'_0} p_i$, and let $I = \{i \in \mathbb{H} \mid p_i \ge q\}$.

Note that the total probability mass of $\mathbb{H} \setminus I$ is at most $k \cdot q = O(p(\mathbb{T}))$. Also, for all $i \in I$ and all ℓ, we have $p_i \ge 2 \sum_{j \in G_\ell} p_j$. By Corollary 20, after γ requests to i, the probability any page in G_ℓ ever has count exceeding that of i is $2^{-\gamma} = 1/\mathrm{poly}(k)$. By the same argument, the probability any page in $\{k+1, \dots, 2k\}$ exceeds the count of i after γ requests to i is also $1/\mathrm{poly}(k)$. By a union bound, no page in the tail exceeds the count of i after γ requests to i with probability $1 - \mathrm{poly}(k)$, and hence the expected number of requests to i before this condition holds is at most 2γ.

Case 2. Suppose now that $n_{\max} \le k$, so $\gamma = 16 \log n_{\max}$. Then let $L_\ell = \{j \in \mathbb{T} \mid p_i \in [p_k 2^\ell, p_k 2^{\ell-1})\}$. By definition $|L_\ell| \le n_{\max}$. For every $i \in \mathbb{H}$ and every $\ell \ge 1$, by Corollary 20 after γ requests to i, a page j in L_ℓ will have count exceeding i with probability no more than $2^{-h\gamma/8} = (n_{\max})^{-2\ell}$. By a union bound, no page in L_ℓ has count exceeding i with probability $(n_{\max}^{-\ell}) \le 2^{-\ell-1}$, and hence by another union bound, no page in $\mathbb{T}$ with probability more than $p_i/2$ will have count exceeding i with probability at least $1/2$. Once again, this means the expected number of requests to i before this condition holds is 2γ. ◁

E Computing Subset Entropy

We recall the definition of k-subset entropy (2.1). We show the following lemma, which immediately implies an efficient algorithm for computing $\mathscr{H}(\mathcal{D}, k)$ (we call this the interval property of $\mathscr{H}(\mathcal{D}, k)$):

▶ **Lemma 24** (Interval Property of $\mathscr{H}(\mathcal{D}, k)$)**.** *Consider a discrete distribution $\mathcal{D} : [n] \to [0,1]$ such that $\mathcal{D}(i) \ge \mathcal{D}(j)$ for any $i \ge j$. Let $k \le n$ be any positive integer. Then, there is always a subset $S \subseteq [n], |S| \le k$ that maximizes $\mathrm{H}(\mathcal{D}_S)$ among subsets of $[n]$ of cardinality at most k which is of the form $S = \{r, r+1, \dots, t\}$.*

Next, we prove this lemma. An important parameter in our proof will be the weighted geometric mean defined as $\tau = \prod_{i \in S} p_i^{q_i}$.

▷ Claim 25. The weighted geometric mean $\tau = \prod_{i \in S} p_i^{q_i}$ satisfies the following condition:

$$\frac{\partial \mathrm{H}(S)}{\partial p_j} \begin{cases} < 0 & \text{if } p_j > \tau \\ = 0 & \text{if } p_j = \tau \\ > 0 & \text{if } p_j < \tau. \end{cases}$$

Proof. Let us use the notation $p_i := \mathcal{D}(i)$, i.e., $p_1 \ge p_2 \ge \cdots \ge p_n$. We have:

$$\mathrm{H}(S) = -\frac{\sum_{i \in S} p_i \log \frac{p_i}{\sum_{i \in S} p_i}}{\sum_{i \in S} p_i} = -\frac{\sum_{i \in S} p_i \log p_i}{\sum_{i \in S} p_i} + \log\left(\sum_{i \in S} p_i\right).$$

Note that the entropy of the conditional distribution on S given by $\mathrm{H}(\mathcal{D}_S)$ is a function of the values $p_i, i \in S$. We consider the partial derivative of this function with respect to any specific $p_j, j \in S$:

$$\begin{aligned} \frac{\partial \mathrm{H}(S)}{\partial p_j} &= -\frac{\left(\sum_{i \in S} p_i\right)(1 + \log p_j) - \sum_{i \in S} p_i \log p_i}{\left(\sum_{i \in S} p_i\right)^2} + \frac{1}{\sum_{i \in S} p_i} \\ &= -\frac{\log p_j}{\sum_{i \in S} p_i} + \frac{\sum_{i \in S} p_i \log p_i}{\left(\sum_{i \in S} p_i\right)^2} = \frac{\frac{\sum_{i \in S} p_i \cdot \log p_i}{\sum_{i \in S} p_i} - \log p_j}{\sum_{i \in S} p_i} = \frac{\sum_{i \in S} q_i \log p_i - \log p_j}{\sum_{i \in S} p_i}, \end{aligned}$$

where $q_i := \frac{p_i}{\sum_{i \in S} p_i}$. Recall that $\tau = \prod_{i \in S} p_i^{q_i}$, i.e., $\log \tau = \sum_{i \in S} q_i \log p_i$. The claim follows. ◁

Next, we state some simple properties of the weighted geometric mean τ.

▷ Claim 26. The weighted geometric mean τ over a set S does not change if τ is added to S.

Proof. Recall that $\tau = \prod_{i \in S} p_i^{q_i}$ where $q_i = p_i/p$, where $p = \sum_{i \in S} p_i$. Let q'_i be the weighted geometric mean if we include τ, i.e.,

$$q'_i = \begin{cases} p_i/p+\tau & \text{if } i \in S \\ \tau/p+\tau & \text{if } p_i = \tau. \end{cases}$$

The new weighted geometric mean τ' is given by

$$\tau' = \left(\prod_{i \in S} p_i^{q'_i}\right) \cdot \tau^{\tau/p+\tau} = \left(\prod_{i \in S} p_i^{p_i/p+\tau}\right) \cdot \tau^{\tau/p+\tau} = \left(\prod_{i \in S} p_i^{p_i/p}\right)^{p/p+\tau} \cdot \tau^{\tau/p+\tau} = \tau^{p/p+\tau} \cdot \tau^{\tau/p+\tau} = \tau. \quad \triangleleft$$

▷ Claim 27. Consider two sets $S_1 = S \cup \{p_{\max}\}, S_2 = S \cup \{p_{\min}\}$ such that for any $i \in S$, we have $p_{\min} \le p_i \le p_{\max}$. Let τ_1, τ_2 be the weighted geometric means of S_1, S_2 respectively. Then, $\tau_1 \ge \tau_2$.

Proof. Let $p = \sum_{i \in S} p_i$. We have

$$\frac{\tau_1}{\tau_2} = \frac{\prod_{i \in S} p_i^{p_i/p+p_{\max}}}{\prod_{i \in S} p_i^{p_i/p+p_{\min}}} \cdot \frac{p_{\max}^{p_{\max}/p+p_{\max}}}{p_{\min}^{p_{\min}/p+p_{\min}}} \ge \prod_{i \in S} p_i^{\frac{p_i(p_{\min}-p_{\max})}{(p+p_{\max})(p+p_{\min})}} \cdot p_{\max}^{\frac{p(p_{\max}-p_{\min})}{(p+p_{\max})(p+p_{\min})}}$$

$$= \prod_{i \in S} \left(\frac{p_{\max}}{p_i}\right)^{\frac{p_i(p_{\max}-p_{\min})}{(p+p_{\max})(p+p_{\min})}} \ge 1. \quad \triangleleft$$

Now, we are ready to prove Lemma 24.

Proof of Lemma 24. Let $S_{\max}$ denote a subset that maximizes $\mathrm{H}(\mathcal{D}_S)$. Let $p_{\min}, p_{\max}$ respectively denote the minimum and maximum values in $S_{\max}$. Assume for contradiction that there is some $q \notin S_{\max}$ such that $p_{\max} < q < p_{\min}$. We will show that $\max\left(\mathrm{H}(S_{\max} \cup \{q\} \setminus \{p_{\max}\}), \mathrm{H}(S_{\max} \cup \{q\} \setminus \{p_{\min}\})\right) \ge \mathrm{H}(S_{\max})$. Clearly, this would establish the lemma.

We consider two continuous processes. The first is to decrease the value $p_{\max}$. As we do so, it might not remaining the maximum value in set $S_{\max}$ but we still denote the dynamic value by $p_{\max}$. Note that the value of the weighted geometric mean of $S_{\max}$, denoted τ, also changes; the termination condition we define below is with respect to the current value of the geometric mean and not its original value. This process ends when either of the following two events happens:

- either, $p_{\max}$ decreases to the current value of the weighted geometric mean τ.
- or, $p_{\max}$ decreases all the way to q while always staying above the (dynamic) weighted geometric mean τ. Then, by Claim 25, the value of $\mathrm{H}(S_{\max})$ is non-decreasing throughout this entire process. Therefore, we have replaced $p_{\max}$ by q and not decreased $\mathrm{H}(S_{\max})$, which is a contradiction.

Next, we consider the second process. This is similar to the first process, except that we now increase $p_{\min}$ instead of decreasing $p_{\max}$. As earlier, as we increase the value of $p_{\min}$, it might not remaining the minimum value in set $S_{\max}$ but we still denote the dynamic value by $p_{\min}$. Also, as in the previous process, the value of the weighted geometric mean τ changes during this process, and the termination condition we define below is with respect to the current value of the geometric mean and not its original value. This process ends when either of the following two events happens:

- either, $p_{\min}$ increases to the current value of the weighted geometric mean τ.
- or, $p_{\min}$ increases all the way to q while always staying below the (dynamic) weighted geometric mean τ. Then, by Claim 25, the value of $\mathrm{H}(S_{\max})$ is non-decreasing throughout this entire process. Therefore, we have replaced $p_{\min}$ by q and not decreased $\mathrm{H}(S_{\max})$, which is a contradiction.

We are left with the case that in both processes, the first event happened. Let τ_1, τ_2 be the final values of τ in these two processes respectively. For the first event to happen in the first process, we must have $q < \tau_1$; similarly, for the first event to happen in the second process, we must have $q > \tau_2$. Thus

$$\tau_2 < q < \tau_1. \tag{E.17}$$

By *Claim* 26, the value of τ_1 is the weighted geometric mean of the set $S_{\max} \setminus \{p_{\max}\}$; similarly, the value of τ_2 is the weighted geometric mean of the set $S_{\max} \setminus \{p_{\min}\}$. Therefore, by Claim 27, $\tau_1 < \tau_2$. This contradicts (E.17). ◀

F Lower Bound on LRU

▶ **Theorem 28.** *There are distributions $\mathcal{D}$ such that the cost of LRU is $\Omega(p(\mathbb{T})) \cdot \mathscr{H}$.*

Proof. Let $\mathcal{D}$ be a distribution that has one page with probability $\varepsilon = 4/(k \log k)$, and k pages with probability $(1-\varepsilon)/k$. By a Markov bound, the ε probability page appears by time $2/\varepsilon = k \log k/2$ with probability at least $1/2$. Furthermore, by lower bounds on the coupon collector time (see, e.g., [28, pages 60-63]) the last page of the top k appears before time $9/10 \cdot k \log k$ with probability less than $1/4$. Hence, with probability at least $1/4$, the algorithm pays $(1-\varepsilon)/k = \Omega(p(\mathbb{T}) \cdot \mathscr{H})$. ◀

This does not rule out that LRU is $\mathscr{H}$-competitive, since this bound is witnessed by a distribution for which $p(\mathbb{T}) \approx \mathsf{opt}$, but it means a different proof technique than ours is required.

Socially Fair Clustering: Parameterized Approximation and Local Search

Aditya Anand
Computer Science and Engineering, University of Michigan, Ann Arbor, MI, USA

Yury Makarychev
Toyota Technological Institute at Chicago (TTIC), IL, USA

Liren Shan
Toyota Technological Institute at Chicago (TTIC), IL, USA

Abstract

We study the Socially Fair Clustering problem introduced by Abbasi, Bhaskara, and Venkatasubramanian [2] and by Ghadiri, Samadi, and Vempala [14], along with its extension, the (p, q)-Socially Fair Clustering problem. This problem generalizes k-median and k-means to settings where data points are partitioned into ℓ groups, and the goal is to find a fair clustering that is simultaneously good for all groups. We present several algorithms for this problem.

1. For ℓ_p-Socially Fair Clustering, we give the first constant-factor FPT-approximation parameterized by the number of groups ℓ, resolving the open question raised by Ghadiri, Singh, and Vempala [15]. Our main ingredient is a new algorithm for closing additional centers in parameterized time inspired by local search.
2. We then turn to the more general (p, q)-Socially Fair Clustering problem. The known algorithm for this problem, proposed by Chlamtáč, Makarychev, and Vakilian [13], achieves a very good approximation but is complex, slow and difficult to implement. We analyze the performance of a simple local search algorithm and show that it provides an $O(q)$-approximation in the worst case.
3. Finally, we design approximation algorithms for the facility location variant of the problem, where the number of facilities (centers) is not fixed in advance, and opening each facility incurs an opening cost. Unlike in previous work, we do not assume these opening costs are the same for all groups.

2012 ACM Subject Classification Theory of computation → Facility location and clustering; Theory of computation → Fixed parameter tractability

Keywords and phrases Socially fair clustering, Approximation algorithms, Fixed-parameter tractability, Local search, Facility location, k-median and k-means

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.3

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2608.30026

Funding *Yury Makarychev*: NSF award ECCS-2216899

1 Introduction

1.1 Overview

Today, computers are widely used to analyze and classify data, generate recommendations, and make decisions that affect many aspects of people's lives. Traditionally, algorithms have focused on optimizing the overall quality of a solution. As a result, their decisions may be unfair to specific individuals or groups. Given the significance of these decisions, addressing fairness in algorithmic decision-making has become essential.


© Aditya Anand, Yury Makarychev, and Liren Shan;
licensed under Creative Commons License CC-BY 4.0

The k-median and k-means objectives are widely used in clustering and typically yield solutions that perform well for most data points. However, some points may be much farther from their assigned centers than others, and therefore incur disproportionately high costs. This is especially problematic when data points represent people, and those bearing the highest costs disproportionately belong to minority groups. Consider, for example, a facility location problem modeled using k-median: we wish to open k facilities (e.g., hospitals or grocery stores) in a city to serve all residents effectively. The goal is to minimize the sum of distances from each person (data point) to their nearest facility (center): $\sum_{u \in X} d(u, C)$. If a minority group resides in an isolated neighborhood, it is possible that none of the facilities will be placed nearby, leaving that group underserved.

To address this issue, [2] and [14] independently introduced the *Socially Fair Clustering* problem. This problem was also previously studied by [6] in the context of clustering under uncertainty. In the fair clustering setting, the n data points belong to ℓ groups $G_1, \ldots, G_\ell \subseteq X$, representing minority or otherwise protected populations (the groups may overlap and do not have to cover X). The goal is to find a clustering that performs well for all groups simultaneously. It is convenient to assign weights to data points: for group i, let $w_i(u)$ denote the weight of point u, where $w_i(u) > 0$ if $u \in G_i$ and $w_i(u) = 0$ otherwise; one natural choice is $w_i(u) = 1/|G_i|$ for $u \in G_i$.

The ℓ_p-Socially Fair Clustering problem, referred to as ℓ_p-Clustering for brevity, asks for a set of k centers C from a set of potential centers $F \subseteq X$ that minimizes

$$\max_{i \in \{1,\ldots,\ell\}} \left(\sum_{u \in X} w_i(u) d(u, C)^p \right)^{1/p}, \tag{1}$$

ensuring that the solution performs well across all groups. In other words, we compute the p-norm of distances $\left(\sum_{u \in X} w_i(u) d(u, C)^p\right)^{1/p}$ for each group G_i, and the objective aggregates these by taking a maximum (many papers define the objective as the p-th power of ours; the two formulations are essentially equivalent: an α-approximation for our objective corresponds to an α^p-approximation for the alternative, and vice versa).

A more general variant of the problem, known as (p, q)-Socially Fair Clustering with $1 \leq p \leq q \leq \infty$, employs a different aggregation function.[1] In this formulation, we construct a vector whose i-th coordinate represents the p-norm of the distances for group i, and then take the ℓ_q norm of this vector:

$$\left(\sum_{i=1}^{\ell} \left(\sum_{u \in X} w_i(u) d(u, C)^p \right)^{q/p} \right)^{1/q}. \tag{2}$$

We refer to this problem simply as (p, q)-Clustering. When $q = \infty$, the objective in (2) reduces to that in (1); thus, ℓ_p-Clustering and (p, ∞)-Clustering are equivalent. Moreover, the $(p, \max\{p, \log \ell\})$-objective approximates the ℓ_p-objective within a constant factor. This follows from the more general fact that the ℓ_q and ℓ_r norms of a d-dimensional vector

[1] We assume that $q \geq p$, since otherwise the objective would favor *unfair* clusterings. For example, let $p = 2q$ and $\ell = 2$. Suppose there are two candidate solutions: in the first, each group has p-norm distance 1, while in the second, the p-norms of the distances for the two groups are $2^{1/p}$ and 0 (where the p-norm distance of group i is given by $\left(\sum_{u \in X} w_i(u) d(u, C)^p\right)^{1/p}$). The corresponding aggregated objectives are $(1^{1/2} + 1^{1/2})^{1/q} = 2^{1/q}$ and $(2^{1/2} + 0^{1/2})^{1/q} = 2^{1/(2q)}$. Thus, we would prefer the second solution, even though it is clearly less fair.

are within a factor of $d^{|\frac{1}{q}-\frac{1}{r}|}$ of each other, which can be shown by a simple computation. Hence in all our results for (p,q)-Clustering, we obtain results for ℓ_p-Clustering by setting $q = \max\{p, \log \ell\}$.

1.2 Known results

The problem has been extensively studied since its introduction. We begin by summarizing previously published approximation algorithms. [6] gave an $O(\log n + \log \ell)$-approximation for ℓ_1-Clustering. [2] and [14] provided $O(\ell^{1/p})$-approximation algorithms for ℓ_p-Clustering with $p \in \{1, 2\}$. Additionally, [2] presented a bicriteria approximation algorithm that achieves a constant-factor approximation by opening $O(k)$ centers.

The approximation factor was later improved by [22], who obtained an $O\left(\frac{\log \ell}{\log\log \ell}\right)^{1/p}$-approximation. This approximation factor is optimal under the assumption that $NP \not\subseteq \bigcap_{\delta>0} DTIME(2^{n^\delta})$, as was shown by [10]. Finally, [13] studied the (p,q)-Clustering problem and gave an $O\left(\frac{q}{\ln(1+q/p)}\right)^{1/p}$-approximation for it. This approximation factor matches that for ℓ_p-clustering when $q = \log \ell$.

Another line of research has focused on designing fixed-parameter tractable (FPT) algorithms for ℓ_p-Clustering. In a significant result, [16] settled the approximability of ℓ_p-Fair Clustering for FPT algorithms parameterized by k by designing a $3+\varepsilon$-approximation algorithm and proving a matching lower bound of $3-\varepsilon$. They also established lower bounds of $1 + 2/e - \varepsilon$ and $\sqrt{1+8/e} - \varepsilon$ for the approximability of ℓ_1 and ℓ_2-Clustering when parameterized by both k and ℓ.

[15] observed that in practice, ℓ is typically a small constant, while k may be very large. They presented a polynomial-time algorithm achieving a $5 + 2\sqrt{6}$ bicriteria approximation by opening at most $k + \ell$ centers. Building on the bicriteria result mentioned above, they also designed algorithms with running times $n^{O_p(\ell^2)}$ and $k^\ell \mathrm{poly}(n)$, achieving approximation factors of $5 + 2\sqrt{6} + \varepsilon$ and $15 + 6\sqrt{6} + \varepsilon$, respectively. The latter is an FPT algorithm parameterized by both k and ℓ. They then raised the question of whether there exists a FPT algorithm achieving a constant-factor approximation parameterized solely by ℓ. This question remained open until our current work.

We also note that the problem admits a $(1+\varepsilon)$-approximation when parameterized by k for certain special classes of metric spaces, as shown by [1]. In many real-world applications, the number of centers – typically referred to as facilities in this context – is not fixed in advance. Instead, opening each facility incurs a cost. As is standard, we refer to this setting as the Facility Location problem. [2] presented a 4-approximation algorithm for the Facility Location variant of ℓ_1-Clustering, assuming homogeneous costs across groups (i.e., the cost of opening a facility is shared equally among all groups).

1.3 Other related work

The theme of fair clustering has received a lot of attention in recent years. Some other notions of fairness that have been adopted include balancedness [11], no over-representation [4], fair hierarchical clustering [3], individual fairness [20], pairwise fairness [9, 8]. Many of these incorporate constraints on clusters rather than enforcing fairness through a new objective, as is the case for Socially Fair Clustering.

1.4 Our results

1.4.1 FPT Algorithm parameterized by ℓ

In this paper, we address several important open problems in the field. First, we resolve the open question posed by [15] by presenting a constant-factor approximation FPT algorithm for ℓ_p-Clustering, parameterized by the number of groups ℓ.

▶ **Theorem 1.** *For every $\varepsilon > 0$, there exists a $(17 + 6\sqrt{6} + \varepsilon)$-approximation algorithm for ℓ_p-Clustering that runs in time $f(\ell, p, \frac{1}{\varepsilon}) \cdot n^{O(1)}$.*

We prove this result by designing a new algorithm that reduces the number of centers in fixed-parameter time.

▶ **Theorem 2.** *There exists an algorithm that, given an instance of ℓ_p-Clustering, a c-approximate solution C with $k + t$ centers, and a parameter $\varepsilon > 0$, returns a $(3c + 2 + \varepsilon)$-approximate solution with k centers. The algorithm runs in time $f(\ell, p, t, \frac{1}{\varepsilon}) \cdot n^{O(1)}$.*

Combining Theorem 2 with the following result of [15], we immediately obtain Theorem 1, thereby answering their open question.

▶ **Theorem 3** ([15])**.** *There is a bicriteria polynomial-time algorithm for ℓ_p-Clustering that finds a $(5 + 2\sqrt{6})$-approximate solution with $k + \ell$ centers.*

Additionally, in the full version of this paper, we strengthen Theorem 3 in the important case of ℓ_1-Clustering by presenting a local-search algorithm that finds a $(3+\varepsilon)$-approximation with $k + f(\ell, \frac{1}{\varepsilon})$ centers. This yields a variant of the algorithm from Theorem 1 with a $(11 + \varepsilon)$-approximation for ℓ_1-Clustering.

1.4.2 Local Search Algorithm for (p, q)-Clustering

The second theme of our paper is the design of a more efficient and implementable algorithm for (p, q)-Clustering. The algorithm in [13] solves an exponential-size convex relaxation using the round-or-cut framework, which is impractical and difficult to implement. In contrast, local search algorithms are well known for their efficiency and ease of implementation. Previously, [7] showed how to obtain a $(5 + \varepsilon)$-approximation for k-median using a single-swap local search algorithm, and a $(3 + \varepsilon)$-approximation using a multiple-swap variant. This result was generalized by [17], who gave an $O(p)$-approximation for the single-group case of ℓ_p-Clustering and showed that this analysis is tight – there are instances where the algorithm returns only an $\Omega(p)$-approximate solution.

We analyze the performance of the single-swap local search algorithm on instances of (p, q)-Clustering with an arbitrary number of groups ℓ, and show that it yields an $O(q)$-approximation. The aforementioned lower bound in [17] implies that our analysis is tight. While our approximation guarantee is worse than those of [22] and [13], it comes close in a particularly important case: ℓ_1-Clustering. For that problem, we obtain an $O(\log \ell)$-approximation by setting $q = \log \ell$, which matches the best known approximation factor of $O(\log \ell / \log \log \ell)$ up to a $\log \log \ell$ factor. Our algorithm is simpler, more implementable, and much faster than the algorithm of [13].[2]

▶ **Theorem 4.** *There is a single-swap local search algorithm for the (p, q)-Clustering problem that gives an $O(q)$-approximation, performs at most $O(kq \log n)$ iterations and runs in time $O_{p,q}(n^2 \ell k^2 \log n)$.*

[2] The running time in [13] is polynomial, but it is not stated explicitly.

1.4.3 (p, q)-Socially Fair Facility Location

Finally, we study the (p, q)-Socially Fair Facility Location problem, an important and practically motivated variant of (p, q)-Clustering. Building on the result of [2] for ℓ_1, we present a $(4 + \varepsilon)$-approximation algorithm for (p, q)-Socially Fair Facility Location with homogeneous costs. We then consider a more general setting with heterogeneous costs, where different groups incur different facility opening costs. In this case, we obtain an $O(q/\log q)$-approximation; in particular, for $p = 1, q = \infty$, this yields an $O(\log \ell/\log\log \ell)$-approximation[3], which almost matches the approximation hardness of $\Omega(\log^{1-\varepsilon} \ell)$ for this problem; see the full version for details.

▶ **Theorem 5.** *For Socially Fair Facility Location with group-homogeneous facility cost, there exists a polynomial-time algorithm that achieves a $(4 + \varepsilon)$-approximation for any $\varepsilon > 0$.*

▶ **Theorem 6.** *For Socially Fair Facility Location with group-heterogeneous facility cost, there exists a polynomial-time algorithm that achieves an $O(q/\log q)$-approximation.*

Organization

Section 3 gives an overview of the proofs of our results. In Section 4, we present our fixed-parameter tractable (FPT) algorithm for ℓ_p-Clustering. In Section 5, we analyze a simple single-swap local search algorithm for (p, q)-Clustering. In Section 6, we describe our algorithm for (p, q)-Socially Fair Facility Location. In the full version of the paper, we give a $(3 + \varepsilon)$-bicriteria local-search algorithm that opens $k + f(\ell)$ centers for ℓ_1-Clustering.

2 Preliminaries

This paper focuses on the ℓ_p and (p, q)-Fair Clustering Problems, referred to below simply as ℓ_p- and (p, q)-Clustering. In these problems, we are given a metric space (X, d), where the points represent data points or clients, and a set of potential center or facility locations $F \subseteq X$. Let n be the number of data points or clients. For each point $u \in X$, we are also given a vector of weights $w(u) \in \mathbb{R}^{\ell}_{\geq 0}$, where the i-th coordinate $w_i(u)$ denotes the weight for group i. We denote the number of groups by ℓ. The goal is to select a subset of k centers $C \subseteq F$ that minimizes objective (1) for ℓ_p-Clustering and (2) for (p, q)-Clustering.

To analyze these objectives, we define the ℓ_p-cost of the clustering (raised to the power p) with respect to a weight function $w_i : X \to \mathbb{R}_{\geq 0}$ and centers C as

$$\mathrm{cost}(w_i, C) = \sum_{u \in X} w_i(u) d(u, C)^p,$$

where $d(u, C) = \min_{c \in C} d(u, c)$. We then define the cost vector induced by weights w and centers C as the ℓ-dimensional vector whose i-th coordinate is $\mathrm{cost}(w_i, C)^{1/p}$. Then the ℓ_p-objective in (1) is the ℓ_∞-norm of the cost vector, while the (p, q)-objective in (2) is its ℓ_q-norm.

We will need the following claim.

▷ Claim 7 (see e.g. Lemma A.1 in [21]). For all non-negative real numbers a and b, and for every $p \geq 1$ and $\varepsilon > 0$, we have

$$(a + b)^p \leq (1 + \varepsilon)^{p-1} a^p + (1 + 1/\varepsilon)^{p-1} b^p,$$

[3] We recall that the $(1, \log \ell)$ objective approximates the $(1, \infty)$ objective up to a constant factor

3 Proof Overview

In this section, we offer a technical overview of the proofs of our main results. We focus primarily on the fixed-parameter tractable (FPT) approximation algorithm for ℓ_p-Clustering parameterized by the number of groups ℓ (Section 4), and then briefly describe the ideas underlying the remaining results.

3.1 FPT approximation for ℓ_p-Clustering (Section 4)

Our goal is to design a constant-factor approximation algorithm for ℓ_p-Clustering that runs in time $f(\ell) \cdot n^{O(1)}$, parameterized only by the number of groups ℓ. As a starting point, we run the algorithm of [15] (see Theorem 3) to get a $(5 + 2\sqrt{6})$-approximate solution with at most $k + \ell$ centers in polynomial time. In the special case of ℓ_1-Clustering, we can alternatively use the local-search-based bicriteria approximation algorithm (see the full version of the paper), which yields a better approximation. Let C denote the resulting set of $k + t$ centers, where $t = \ell$, and let c be the corresponding approximation factor.

Given such a bicriteria solution, the main challenge is to close t centers and reassign the clients assigned to them to other centers, while controlling the cost increase for all groups. We proceed in two steps. First, we show that *existentially* it is possible to close t centers C_1 and reassign every client assigned to a center c in C_1 to the center closest to c in $C \setminus C_1$ so that the cost of the resulting solution is within a constant factor of the cost of C (see Lemma 9 for the exact guarantee). The key idea is that the new connection cost for a client u can be bounded in terms of both the cost of connecting u to its originally assigned center in C_1 and the cost of connecting u to the center serving u in the optimal solution C^*. This observation is important because it allows us to reassign the clients assigned to the t closed centers to at most t centers (that are not closed), and this property is crucial for our algorithm.

However, we cannot use the procedure from Lemma 9 directly, as it requires knowledge of C^*. Thus, to obtain an FPT algorithm, we need to identify the set C_1 without access to the optimal solution. This is achieved using the technique of color coding introduced by [5]. Let $C_2 \subseteq C \setminus C_1$ be the set formed by selecting, for each $c \in C_1$, a closest center to c in $C \setminus C_1$. Note that $|C_2| \leq |C_1| = t$ (each center $c_1 \in C_1$ contributes at most one selected center; and different centers in C_1 may potentially have the same closest center in $C \setminus C_1$). We color all centers in C randomly using two colors, red and blue. Each center is colored red or blue independently with probability $1/2$. Note that with probability at least 2^{-2t}, all centers in C_1 are colored red and all centers in C_2 are colored blue. By running the algorithm $O(2^{2t} \log n)$ times, we obtain a coloring with this property with high probability. We denote the sets of red and blue centers by R and B, respectively.

Now, we introduce the notion of a Z-profile of a center c. Loosely speaking, the Z-profile is a vector in $\mathbb{R}^\ell$ that captures, for each of the ℓ groups, the reassignment cost incurred by closing center c and reassigning all clients assigned to c to the center closest to c in Z. Crucially, the C_2-profiles of all centers in C_1 fully determine the reassignment cost: the reassignment cost vector is simply the sum of the C_2-profiles of all centers in C_1. Importantly, if we were to find another set of centers in R with the same profiles, the reassignment cost would be the same. However, there are two challenges in finding such a set. First, as defined, the number of possible profiles is unbounded. Second, since we do not know the set C_2, we cannot compute the C_2-profiles of centers in R.

To resolve the first issue, we discretize the set of all possible profiles so that their number is bounded by a function of ℓ, t, p, and $1/\varepsilon$. We then "guess" the number of centers in C_1 with each possible profile. Next, we observe that the C_2-profiles and B-profiles of centers in

C_1 are equal, and thus we can use B as a proxy for C_2. We select centers in R according to our guess of the profile counts and obtain a set D_1 of t centers to close. Given the discussion above, it is not hard to see that by closing the centers in D_1 and reassigning clients to their closest centers in B, we obtain a constant-factor approximation.

3.2 Local search for (p, q)-Clustering (Section 5)

Building on the local search algorithms from [7] and [17] for k-median, k-means, and (k, p)-clustering, we design and analyze a local search algorithm for the more general (p, q)-Clustering problem. The general approach is similar to that of [7] and [17], but the analysis is more complex, as it requires fine-grained control over the distribution of swaps (see below).

The algorithm starts with an arbitrary set C of k centers and repeatedly finds and performs a swap $c \to c'$ that replaces one of the centers $c \in C$ with a center c' outside of C. The swaps are chosen to minimize the (p, q)-Clustering objective, and the algorithm terminates when no swap decreases the current potential Φ by more than $\Phi/(8k)$. To prove that the algorithm achieves an $O(q)$-approximation, we show that if the cost of the current solution exceeds the optimal cost by $\Omega(q)$ times, then there must exist an improving swap. To this end, we present a distribution over special swaps that replace centers in C with centers in an unknown optimal solution C^* (The algorithm may choose a swap from the distribution or any other swap improving the objective; we use the distribution only for the analysis). While this approach resembles that of [7] and [17], we require significantly more control over the distribution of swaps to handle the more complex objective. In particular, we use careful linear approximations for the change in the non-linear objective that allow us to show that there is indeed such a distribution over improving swaps.

To obtain these linear approximations, roughly speaking, we can obtain upper and lower bounds on the derivative of the cost function in the desired range. One can then use the mean-value theorem to find a linear approximation of the change in the objective using these bounds. The actual proof is slightly different, and uses convexity arguments along with the mean value theorem. Carefully combining this with the standard linear-objective analysis for k-Median, we show that the algorithm is an $O(q)$ approximation.

3.3 Socially Fair Facility Location (Section 6)

Finally, we study (p, q)-Socially Fair Facility Location. We consider two variants of the problem: facility location with homogeneous opening costs and facility location with heterogeneous opening costs. For the homogeneous case, we use a rounding algorithm inspired by the standard facility location algorithm of [23]. For the heterogeneous case, we replace the greedy step that selects and opens the cheapest facility within a certain ball B_u with a randomized rounding step that opens a facility in B_u with probability proportional to the target probability given by the convex relaxation. We then use Rosenthal's inequality to bound the expected ℓ_q-norm of the solution. We also give a hardness result showing that our algorithm is near-optimal for ℓ_1-Socially Fair Facility Location (the case when $p = 1$, $q = \infty$).

4 FPT algorithm for ℓ_p-socially fair clustering

As discussed in the introduction, we provide an FPT algorithm for ℓ_p-Clustering, parameterized by the number of groups ℓ. The algorithm first finds an approximate solution with $k + t$ centers using Theorem 3. This solution provides only a *bicriteria* approximation. To obtain a *true* approximation, the algorithm closes t centers – using Theorem 2.

We now proceed with the proof of Theorem 2. Given the instance of ℓ_p-Clustering on point set X and metric d, a set of centers C and an assignment $\alpha : X \to C$, we define the cost with respect to the weight function w_i and assignment α as follows:

$$\text{cost}_i(\alpha) = \text{cost}(w_i, \alpha) := \sum_{u \in X} w_i(u) d(u, \alpha(u))^p.$$

Note that the ℓ_p-objective function for the assignment α is then $\max_{i \in [\ell]} \text{cost}_i(\alpha)^{1/p}$. Throughout this section, we fix an optimal set of centers C^* and a corresponding optimal assignment β^*. Let us define OPT to be the optimal assignment ℓ_p-objective, so that $\text{OPT} = \max_i \text{cost}_i(\beta^*)^{1/p}$.

First, we show that given any set C of $k + t$ centers which is a c-approximate solution, indeed there exists a subset of t centers that can be closed such that the resulting subset of k-centers is a $(3c + 2)$-approximate solution. However, besides this, we show a stronger property that our algorithm requires: every client assigned to some center of C (in an optimal assignment of clients to C) that is closed in this process can now be reassigned to one of at most t open centers in C. The next key lemma proves this result. Before we state the lemma, the following definition will be useful.

▶ **Definition 8.** *Given a set of open centers C, an assignment $\alpha : X \to C$ of points to centers in C, and two disjoint subsets $C_1, C_2 \subseteq C$, we define a new assignment α' obtained from α by reassigning clients as follows. For each client $u \in X$, if $\alpha(u) \in C_1$, we find the closest center c_2 in C_2 to $c_1 \equiv \alpha(u)$ and let $\alpha'(u) = c_2$. If $\alpha(u) \notin C_1$, we set $\alpha'(u) = \alpha(u)$ (do not change the assignment).*

We then define $\text{cost}(\alpha, C_1, C_2)$ *to be the vector* $(v_1, v_2, \ldots, v_\ell)$ *with* $v_i = \text{cost}_i(\alpha') - \text{cost}_i(\alpha)$ *for each* $i \in [\ell]$.

Simply put, $\text{cost}_i(\alpha, C_1, C_2)$ measures the increase in cost for group (coordinate) i by re-assigning each point assigned to a center $c_1 \in C_1$ to the center in C_2 closest to c_1. We use the shorthand $\text{cost}(\beta, c_1, C_2)$ to denote $\text{cost}(\beta, \{c_1\}, C_2)$. If β is an optimal assignment, then $\text{cost}(\beta, c_1, C_2)$ is coordinatewise non-negative, since any reassignment can only increase the cost. We also have:

$$\text{cost}(\beta, C_1, C_2) = \sum_{c_1 \in C_1} \text{cost}(\beta, c_1, C_2) \tag{3}$$

We formalize this nearest-neighbor redirection operation in the following lemma, showing that one can close a set C_1 of t centers and reassign all affected clients to a set C_2 of at most t receiving centers while keeping the reassignment cost controlled. The redirection operator in the definition captures exactly what happens when we close centers $C_1 \subseteq C$: for each closed center $c_1 \in C_1$, we send all of its clients to the closest open center $c_2 \in C \setminus C_1$. Let C_2 be the set of such receiving centers.

▶ **Lemma 9.** *Given a c-approximate solution C with $k + t$ centers and a corresponding optimal assignment β of clients to centers in C, there exist disjoint sets $C_1, C_2 \subseteq C$ with $|C_2| \leq |C_1| = t$ such that*

- $\| \text{cost}(\beta, C_1, C_2) \|_\infty \leq ((3c + 2)^p - c^p)\text{OPT}^p$
- $C_2 = \{c_2 : c_2$ *is the closest point from* $C \setminus C_1$ *to* c_1 *for some* $c_1 \in C_1\}$.

Proof. Let $C^* = \{c_1^*, c_2^*, \ldots, c_k^*\}$ be an optimal set of centers, and let β^* be an optimal assignment of clients to the centers in C^*. For each c_i^*, mark $\gamma(c_i^*) \in C$, the closest center to c_i^* in C. Then there are at most k marked centers in C and at least t unmarked ones. Choose any subset of t unmarked centers in C to be the set C_1. For each center of C_1, we take the closest center in $C \setminus C_1$ to obtain the set C_2. Note that this means $|C_2| \leq |C_1|$.

We now show that this choice of C_1, C_2 satisfies the statement of the lemma. Let $c_1 \in C_1$ and let c_2 be the closest center to c_1 in $C \setminus C_1$ (which must be in C_2). Then, for every point $u \in X$ with $\beta(u) = c_1$, we reassign u to c_2.

We wish to bound the reassignment cost of u to c_2. Let $c^* = \beta^*(u)$ be u's assigned center in C^*. By the triangle inequality,

$$\begin{aligned}
d(u, c_2) &\le d(u, c_1) + d(c_1, c_2) \\
&\le d(u, c_1) + d\big(c_1, \gamma(c^*)\big) \text{ (c_2 is the closest center to c_1 in $C \setminus C_1$)} \\
&\le d(u, c_1) + d(c_1, c^*) + d\big(c^*, \gamma(c^*)\big) \\
&\le d(u, c_1) + 2d(c_1, c^*) \text{ (by definition of } \gamma(c^*)) \\
&\le d(u, c_1) + 2\big(d(u, c_1) + d(u, c^*)\big) \text{ (again by triangle inequality)} \\
&\le 3d(u, c_1) + 2d(u, c^*)
\end{aligned}$$

It follows from Claim 7 that for every $\delta > 0$, we get the inequality

$$d(u, c_2)^p \le 3^p d(u, c_1)^p (1+\delta)^{p-1} + 2^p d(u, c^*)^p \left(1 + \frac{1}{\delta}\right)^{p-1}$$

We can now use this to bound the change in cost. By definition, it follows that the contribution of client u to $\mathrm{cost}_i(\beta, C_1, C_2)$ is $w_i(u)\big(d(u, c_2)^p - d(u, c_1)^p\big)$. Now repeating the analysis for each point u, and recalling that $c_1 = \beta(u), c^* = \beta^*(u)$, we obtain

$$\begin{aligned}
\mathrm{cost}_i(\beta, C_1, C_2) \le \sum_{u \in X} w_i(u) \Big(& 3^p (1+\delta)^{p-1} \, d\big(u, \beta(u)\big)^p \\
& + 2^p \Big(1 + \frac{1}{\delta}\Big)^{p-1} d\big(u, \beta^*(u)\big)^p - d(u, \beta(u))^p \Big) \\
\le & \Big((3c)^p (1+\delta)^{p-1} + 2^p \Big(1 + \frac{1}{\delta}\Big)^{p-1} - c^p \Big) \ \mathrm{OPT}^p \quad \text{for each } i \in [\ell],
\end{aligned}$$

where we use the fact that $\sum_{u \in X} w_i(u) d\big(u, \beta^*(u)\big)^p \le \mathrm{OPT}^p$ and $\sum_{u \in X} w_i(u) d\big(u, \beta(u)\big)^p \le c^p \, \mathrm{OPT}^p$, since C is a c-approximate solution.

Now we set $\delta = \frac{2}{3c}$. It then follows that

$$\mathrm{cost}_i(\beta, C_1, C_2) \le \big((3c+2)^p - c^p\big) \, \mathrm{OPT}^p \, . \qquad \blacktriangleleft$$

Next, we describe how to find such sets in FPT time. We will use the color coding technique of [5]. We remark that while our algorithm is stated as a randomized algorithm for simplicity, it can be easily derandomized using standard techniques (see, for example, [12, Lemma 1]).

▶ **Lemma 10.** *Given a c-approximate solution C with $k + t$ centers and a corresponding optimal assignment β of clients to centers in C, one can find disjoint sets D_1, D_2 with $|D_2| \le |D_1| = t$ satisfying $\|\mathrm{cost}(\beta, D_1, D_2)\|_\infty \le \big((3c+2)^p - c^p + \varepsilon^p\big) \, \mathrm{OPT}^p$ in time $f(\ell, t, p, \frac{1}{\varepsilon}) n^{O(1)}$.*

Proof. Let $C_1, C_2 \subseteq C$ be the (unknown) subsets of C guaranteed by Lemma 9. We use color coding. Randomly and independently color each center in C red or blue, each with probability $1/2$. Call a coloring *good* if every center of C_1 is colored red and every center of C_2 is colored blue. Then a random coloring is good with probability at least 2^{-2t}. By independently sampling $O(2^{2t} \log n)$ random colorings, we can ensure that we enumerate a good coloring with high probability. Henceforth, assume that we are given a good coloring (we will try all the $O(2^{2t} \log n)$ colorings). Let $R \subseteq C$ be the red-colored centers, and $B \subseteq C$ be the blue-colored centers in this good coloring.

For a set $Z \subseteq C$, we say that the Z-profile of $c \in C$ is $(i_1, i_2, \ldots, i_\ell)$ if the m-th coordinate of $\text{cost}(\beta, c, Z)$ is in the interval $\left[\frac{(i_m-1)\varepsilon^p}{t}\,\text{OPT}^p, \frac{i_m\varepsilon^p}{t}\,\text{OPT}^p\right]$ for each $m \in [\ell]$. Here $i_1, i_2 \ldots i_\ell$ are positive integers. Let $\mathcal{P}$ be the set of profiles with all $i_j \le t(\frac{3c+2}{\varepsilon})^p$. Then $|\mathcal{P}| \le (t(\frac{3c+2}{\varepsilon})^p)^\ell$. Note that the C_2-profile of every $c_1 \in C_1$ lies in $\mathcal{P}$. This is the case because, for every coordinate $i \in [\ell]$, (1) $\text{cost}_i(\beta, c_1, C_2) \ge 0$, since β is an optimal assignment, and (2) $\text{cost}_i(\beta, c_1, C_2) \le \text{cost}_i(\beta, C_1, C_2) \le \|\text{cost}(\beta, C_1, C_2)\|_\infty \le (3c+2)^p\,\text{OPT}^p$ by (3) and Lemma 9.

For each profile $P \in \mathcal{P}$, guess the number of centers in C_1 with C_2-profile P. Let n_P denote this number for each P. Then $\sum_{P\in\mathcal{P}} n_P = |C_1| = t$. The number of guesses we need to make is at most $|\mathcal{P}|^t \le \left(t(\frac{3c+2}{\varepsilon})^p\right)^{\ell t}$.

Consider $c_1 \in C_1$ and let c_2 be the closest point to c_1 in $C \setminus C_1$. Note that $c_2 \in C_2$ (per item 2 in Lemma 9) and c_2 is also the closest point to c_1 in B, since $c_2 \in C_2 \subseteq B \subseteq C \setminus C_1$. Therefore, the C_2- and B-profiles of c_1 are the same. In particular, there are n_P centers with B-profile P in C_1. Since all centers in C_1 are red (recall that we assumed that the coloring is good), there are at least n_P red centers with B-profile P.

Now for each $P \in \mathcal{P}$, we choose exactly n_P red centers whose B-profile is P to form the set D_1[4]. Once we choose D_1, we let D_2 be the set of all centers $d_2 \in B$ such that d_2 is the closest center in B to some $d_1 \in D_1$. This concludes the construction of the sets D_1 and D_2.

▶ **Lemma 11.** $\|\text{cost}(\beta, D_1, D_2)\|_\infty \le ((3c+2)^p - c^p + \varepsilon^p)\,\text{OPT}^p$.

Proof. There is a one-to-one map $f : C_1 \to D_1$ such that the C_2-profile of c_1 and the B-profile of $d_1 = f(c_1)$ are the same for every $c_1 \in C_1$.

It follows that for every $d_1 \in D_1$ and every coordinate $i \in [\ell]$,

$$\text{cost}_i(\beta, d_1, D_2) = \text{cost}_i(\beta, d_1, B) \le \text{cost}_i(\beta, f^{-1}(d_1), C_2) + \frac{\varepsilon^p}{t}\,\text{OPT}^p\,.$$

The term $\frac{\varepsilon^p}{t}\,\text{OPT}^p$ is exactly the length of an interval in the definition of profiles. Summing this upper bound on $\text{cost}(\beta, d_1, B)$ over all $d_1 \in D_1$ and using (3), we get

$$\begin{aligned}
\text{cost}_i(\beta, D_1, D_2) &= \sum_{d_1\in D_1} \text{cost}_i(\beta, d_1, D_2) \\
&\le \sum_{d_1\in D_1} \left(\text{cost}_i(\beta, f^{-1}(d_1), C_2) + \frac{\varepsilon^p}{t}\,\text{OPT}^p\right) \\
&\le \text{cost}_i(\beta, C_1, C_2) + \varepsilon^p\,\text{OPT}^p \le \left((3c+2)^p - c^p + \varepsilon^p\right)\text{OPT}^p.
\end{aligned}$$

◀

◀

Proof of Theorem 2. Let β be an optimal assignment for the center set C. Invoke Lemma 10 to obtain the sets D_1, D_2. After obtaining the sets D_1, D_2, we close the centers D_1. Since $|D_1| = t$, it follows that we obtain a solution with at most k centers.

We have $\|\text{cost}(\beta)\|_\infty \le c^p\,\text{OPT}^p$, and by Lemma 10, the additional cost incurred is at most $\|\text{cost}(\beta, D_1, D_2)\|_\infty \le ((3c+2)^p - c^p + \varepsilon^p)\,\text{OPT}^p$. Thus the ℓ_p-objective of the resulting solution with k centers is at most $\left((3c+2)^p - c^p + \varepsilon^p + c^p\right)^{1/p}\text{OPT} \le (3c+2+\varepsilon)\,\text{OPT}$ where in the final step we use the inequality $(a+b)^{1/p} \le a^{1/p} + b^{1/p}$, which holds for every $p \ge 1, a, b \ge 0$. ◀

[4] While we do not know OPT, we can guess OPT^p up to a factor of $(1+\varepsilon)$, but we omit this detail in the proof for the sake of simplicity

By Theorem 3 of [15], we can find a $c = 5 + 2\sqrt{6}$-approximate solution with $k + \ell$ centers in polynomial time. By Lemma 10, we can close ℓ centers in this bicriteria approximation solution to get a solution with k centers in time $\left(\ell(\frac{3c+2}{\varepsilon})^p\right)^{\ell^2} n^{O(1)}$. This proves Theorem 1.

5 Local search algorithm for (p, q)-clustering

5.1 Local search with scalar costs

Following [7] and [17], we first revisit the local search algorithm for (k, p)-clustering. We will later apply the results of this section to analyze the performance of the local search algorithm for (p, q)-Clustering.

Let (X, d) be a metric space, and let $w(u)$ be non-negative vertex weights. We consider one step of the local search algorithm. Let $C = \{c_1, \dots, c_k\}$ be the current set of k centers, and $C^* = \{c_1^*, \dots, c_k^*\}$ be a target set. Later, C^* will be an optimal solution for the (p, q)-objective, but for now, we make no assumptions about it.

We define the cost of C as

$$\operatorname{cost}(w) = \operatorname{cost}(w, C) = \sum_{u \in X} w(u) d(u, C)^p,$$

and the cost of C^* as

$$\operatorname{cost}^*(w) = \operatorname{cost}(w, C^*) = \sum_{u \in X} w(u) d(u, C^*)^p.$$

Consider a swap operation $c_a \to c_b^*$ that replaces c_a with c_b^*. We denote the resulting set of centers by

$$C_{ab} = (C \setminus \{c_a\}) \cup \{c_b^*\},$$

and its cost by $\operatorname{cost}_{ab}(w) = \operatorname{cost}(w, C_{ab}) = \sum_{u \in X} w(u) d(u, C_{ab})^p$.

Consider a probability distribution $\mathcal{D}$ on the set of swaps $c_a \to c_b^*$ (equivalently, on $[k] \times [k]$). The expected cost after a random swap drawn from $\mathcal{D}$ is $\mathbb{E}_{(a,b)\sim\mathcal{D}}[\operatorname{cost}_{ab}(w)]$.

We derive the following guarantees for local search. Conceptually, the proof closely follows the analyses in [7] and [17]. However, the bounds we require are more intricate, as we need finer control over the distribution of swaps. We prove this lemma in Section 5.4.

▶ **Lemma 12.** *Let (X, d) be a metric space, and let C and C^* be two sets of centers. There exists a probability distribution $\mathcal{D}$ over swaps such that the following holds. For every $p \geq 1$, weights w, parameters $M \geq m > 0$, and coefficients $\xi_{ab} \in [m, M]$, we have:*

1. *For $\alpha = \frac{10Mp}{m}$, the following inequality holds.*

$$\mathbb{E}_{\mathcal{D}}[\xi_{ab} \cdot (\operatorname{cost}_{ab}(w) - \operatorname{cost}(w))] \leq \frac{m}{2k}(\alpha^p \operatorname{cost}^*(w) - \operatorname{cost}(w)) \tag{4}$$

2. *We also have:*

$$\mathbb{E}_{\mathcal{D}}[\xi_{ab} \cdot (\operatorname{cost}_{ab}(w) - \operatorname{cost}(w))] \leq \frac{(1 + 2^{2p})M}{k}(\operatorname{cost}^*(w) + \operatorname{cost}(w)) \tag{5}$$

3. *Finally, for every swap $c_a \to c_b^*$ in the support of $\mathcal{D}$ and every $\delta_0 \in (0, 1/4)$, we have*

$$\operatorname{cost}_{ab}(w) \leq (1 + \theta_0) \operatorname{cost}^*(w) + (1 + \delta_0) \operatorname{cost}(w), \tag{6}$$

where $\theta_0 = 2^p (5p/(4\delta_0))^{p-1}$.

At a high level, the necessity for such guarantees is as follows. In k-median, the objective is linear, and the effects of the change in cost due to a single swap can be directly accounted for. However, in (p,q)-Clustering, the objective involves two layers of non-linearity (the inner and outer norms). To deal with this non-linearity, we can write the change in the objective for each swap $c_a \to c_b^*$ as a linear change with multipliers ξ_{ab} using the mean value theorem and convexity. Here the multipliers ξ_{ab} are bounded in the range $[m, M]$ for some appropriate choice of m and M. The above lemma then assumes this boundedness of ξ_{ab} and derives guarantees on the change in $\text{cost}(w)$, which in turn bounds the change in the objective.

5.2 Single swap analysis w.r.t. the (p,q)-objective

Now we consider the local search algorithm for (p,q)-Clustering – the goal is to minimize the following objective function:

$$\left(\sum_{i=1}^{\ell}\left(\sum_{u\in X} w_i(u)d(u,C)^p\right)^{q/p}\right)^{1/q} = \left(\sum_{i=1}^{\ell}\text{cost}(w_i,C)^{q/p}\right)^{1/q}$$

From now on, let C^* be an optimal solution for the (p,q)-Clustering problem. We assume that swap $c_a \to c_b^*$ is sampled from distribution $\mathcal{D}$ constructed in Lemma 12. Define $r = q/p$ and $\varphi(t) = t^r$, and

$$z_i = \text{cost}(w_i), \qquad z_i' = \text{cost}_{ab}(w_i), \qquad z_i^* = \text{cost}^*(w_i)$$
$$\Phi = \sum_{i=1}^{\ell}\varphi(z_i), \qquad \Phi_{ab} = \sum_{i=1}^{\ell}\varphi(z_i'), \qquad \Phi^* = \sum_{i=1}^{\ell}\varphi(z_i^*)$$

Note that $(\Phi^*)^{1/q}$ is the cost of the optimal solution, $\Phi^{1/q}$ is the cost of the current solution, and $\Phi_{ab}^{1/q}$ is the cost of the solution after a swap $c_a \to c_b^*$. Also, let

$$\Delta_i = \mathbb{E}\left[\varphi(z_i') - \varphi(z_i)\right].$$

Fix a coordinate i. Now we get an upper bound for $\varphi(z_i')$ by approximating φ with a piecewise linear function. Let $\hat{z}_i$ be the maximum of z_i' over all swaps in the support of $\mathcal{D}$ (we will obtain an upper bound on $\hat{z}_i$ later using Lemma 12, item 3). Let $M = \frac{\varphi(\hat{z}_i)-\varphi(z_i)}{\hat{z}_i - z_i}$ and $m = \varphi(z_i)/z_i = z_i^{r-1}$. Note that $M \le \varphi'(\hat{z}_i)$ when $\hat{z}_i > z_i$, by the Mean Value Theorem and monotonicity of φ'. Since φ is convex, we have (see Figure 1)

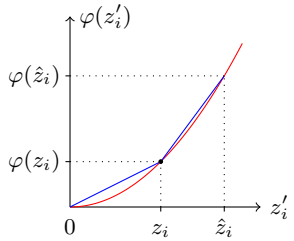


Figure 1 We approximate $\varphi(z_i')$ on the interval $[0, \hat{z}_i]$ using a piecewise linear function consisting of two segments: the first with slope m, and the second with slope M.

$$\varphi(z_i') \le \begin{cases} \varphi(z_i) + m\cdot(z_i' - z_i), & \text{if } z_i' < z_i \\ \varphi(z_i) + M\cdot(z_i' - z_i), & \text{if } z_i' \ge z_i \end{cases}$$

Define coefficients ξ_{ab} as follows

$$\xi_{ab} = \begin{cases} m, & \text{if } z_i' < z_i \\ M, & \text{otherwise} \end{cases}$$

Then $\varphi(z_i') \le \varphi(z_i) + \xi_{ab} \cdot (z_i' - z_i)$, and we have $\Delta_i \equiv \mathbb{E}\left[\varphi(z_i') - \varphi(z_i)\right] \le \mathbb{E}\left[\xi_{ab} \cdot (z_i' - z_i)\right]$.

We are now in a position to use Lemma 12, from which we derive the following key inequality, whose proof is deferred to Section 5.5.

$$\Delta_i \le \frac{1}{4k}\left(\beta^q \varphi(z_i^*) - \varphi(z_i)\right) \tag{7}$$

with $\beta = O(q)$. Adding up the inequality over all coordinates $i \in [\ell]$, we get

$$\mathbb{E}\left[\Phi_{ab} - \Phi\right] = \sum_{i=1}^{\ell} \Delta_i \le \frac{1}{4k}\left(\beta^q \Phi^* - \Phi\right).$$

In particular, this means that there is always a swap $c_a \to c_b^*$, for which

$$\Phi_{ab} - \Phi \le \frac{1}{4k}\left(\beta^q \Phi^* - \Phi\right).$$

5.3 Putting everything together

Consider the local search algorithm that at each iteration finds a swap $c_a \to c'$ with $c_a \in C$ and $c' \in F \setminus C$ that produces the greatest decrease in the potential function Φ, where C is the current set of open centers and F is the candidate facility set. The algorithm terminates when no swap decreases the current potential Φ by more than $\frac{\Phi}{8k}$.

5.3.1 Bounding the number of iterations

Suppose that $\Phi > 2\beta^q \Phi^*$. Then, for the improving swap $c_a \to c_b^*$ identified above, we have

$$\Phi_{ab} - \Phi \le -\frac{\Phi}{8k}$$

and

$$\Phi_{ab} - \Phi^* \le (\Phi - \Phi^*) - \frac{\Phi}{8k} \le \left(1 - \frac{1}{8k}\right)(\Phi - \Phi^*).$$

Thus when the algorithm stops, we must have $\Phi \le 2\beta^q \Phi^*$. Until the algorithm stops, in every iteration, the above inequality holds. It follows that after at most $8k \log_e(\Phi/\Phi^*)$ iterations, the algorithm will find a clustering of cost at most $2^{1/q} \cdot \beta \cdot (\Phi^*)^{1/q}$.

As is standard, by truncating distances and rounding weights, we may assume that all distances and weights are integers bounded polynomially in n. Therefore, the cost of any clustering is polynomial in n; hence, $\log_e(\Phi/\Phi^*)$ is bounded by $O(q \log n)$. We conclude that the algorithm returns a solution of cost at most $O(q)$ OPT in at most $O(qk \log n)$ iterations.

5.3.2 Running time per iteration

In each iteration, we maintain the current set of centers C. For each point u, we maintain its closest and second-closest centers $s_1(u)$ and $s_2(u)$ in C. At the beginning of every iteration, these quantities can be computed in time $O(nk)$. We can compute the change in cost due

to a swap $a \to b$ for each point u using $s_1(u), s_2(u)$ and the distances $d(u,a), d(u,b)$. Then we add these costs for each group in time $O_{p,q}(n\ell)$ and re-compute the potential. Thus, for every swap $a \to b$, we spend time $O_{p,q}(n\ell)$ and there are $O(nk)$ swaps, giving a per-iteration running time of $O(n^2k\ell)$. Since there are $O(kq \log n)$ iterations, the running time is at most $O_{p,q}(n^2\ell k^2 \log n)$. This concludes the proof of Theorem 4.

5.4 Proof of Lemma 12

In this Section, we prove Lemma 12.

Proof.

Defining the distribution of swaps. For each center $c_b^* \in C^*$, let $\gamma(c_b^*)$ denote the nearest center in C (see Figure 2). We partition the centers in C based on the number of preimages under γ:

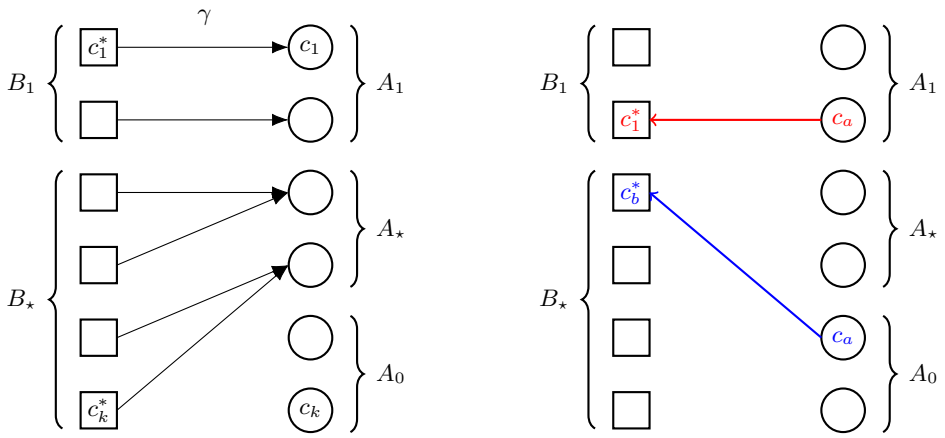


Figure 2 The left figure shows the map γ that maps every optimal center c_i^* to the closest current center c_j. This map defines the partition of the optimal centers into sets B_1 and $B_\star$ and the current centers into sets A_0, A_1, and $A_\star$. The right figure illustrates two possible types of swaps $c_a \to c_b$.

- A_1: centers with exactly one preimage,
- A_0: centers with no preimages,
- $A_\star$: centers with two or more preimages.

Define $B_1 = \gamma^{-1}(A_1)$ and $B_\star = \gamma^{-1}(A_\star)$. We observe:

- γ is a one-to-one map between B_1 and A_1,
- γ maps $B_\star$ onto $A_\star$, with every $c \in A_\star$ having at least two preimages.

Denote $s = |A_1|$. From the observations above, we get $|B_1| = |A_1| = s$ and $|A_\star| \le |B_\star|/2 = (k-s)/2$. Thus, $|A_0| = k - |A_1| - |A_\star| \ge (k-s)/2$. Now we are ready to define a probability distribution $\mathcal{D}$ over swaps.

1. With probability $\frac{s}{k}$, select $c_b^* \in B_1$ uniformly at random and set $c_a = \gamma(c_b^*) \in A_1$,
2. With probability $\frac{k-s}{k}$, select c_a uniformly from A_0 and c_b^* uniformly from $B_\star$.

This yields a random swap $c_a \to c_b^*$. Note that b is uniformly distributed in $[k]$ and therefore the probability that $b = b_0$ is $1/k$ for every fixed $b_0 \in [k]$. In general, a is not uniformly distributed; however, for every fixed $a_0 \in [k]$, the probability that $a = a_0$ is at most $\max(\frac{s}{k} \cdot \frac{1}{|A_1|}, \frac{k-s}{k} \cdot \frac{1}{|A_0|}) \le \max(1/k, 2/k) = 2/k$:

$$\Pr\{a = a_0\} \le 2/k \qquad \text{and} \qquad \Pr\{b = b_0\} = 1/k \tag{8}$$

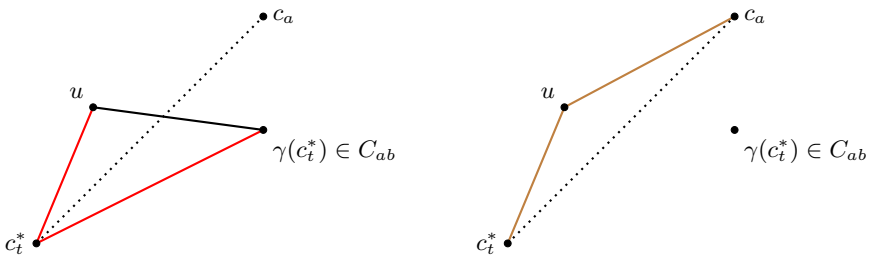


Figure 3 We assign point u to $\gamma(c_t^*)$, which is guaranteed to be in C_{ab}. The distance from u to $\gamma(c_t^*)$ is upper bounded by $d(u, c_t^*) + d(c_t^*, \gamma(c_t^*))$. Further, $d(c_t^*, \gamma(c_t^*)) \le d(c_t^*, c_a)$, since $\gamma(c_t^*)$ is the closest current center to c_t^*. In turn, $d(c_t^*, c_a)$ is upper bounded by $d(c_t^*, u) + d(u, c_a)$.

We will need below that for all $t \neq b$,

$$\gamma(c_t^*) \neq c_a. \tag{9}$$

Indeed, in item 1, $\gamma^{-1}(c_a) = \{c_b^*\}$; in item 2, $\gamma^{-1}(c_a) = \varnothing$. In either case, $c_t^* \notin \gamma^{-1}(c_a)$.

Upper bounding connection cost

Let $x_u = d(u, C)$ denote the distance from u to the closest center in the current solution C, $x'_u = d(u, C_{ab})$ the updated distance after the swap, and $x_u^* = d(u, C^*)$ the distance to the closest center in C^*. Let V_a be the Voronoi cell of c_a in C, and V_b^* the Voronoi cell of c_b^* in C^*. For $u \in V_b^*$,

$$x_u'^p - x_u^p \le x_u^{*p} - x_u^p.$$

For $u \in V_a \setminus V_b^*$, assume $u \in V_t^*$ for some $t \neq b$. By (9), $\gamma(c_t^*) \neq c_a$; thus, $\gamma(c_t^*) \in C_{ab}$ and

$$\begin{aligned} x'_u &= d(u, C_{ab}) \le d(u, \gamma(c_t^*)) \le d(u, c_t^*) + d(c_t^*, \gamma(c_t^*)) \le d(u, c_t^*) + d(c_t^*, c_a) \\ &\le d(u, c_t^*) + \big(d(c_t^*, u) + d(u, c_a)\big) = 2x_u^* + x_u. \end{aligned}$$

Here, we used that $d(c_t^*, \gamma(c_t^*)) \le d(c_t^*, c_a)$, since $\gamma(c_t^*)$ is the nearest center in C to c_t^*.

Applying Claim 7 with $\delta = (1+\varepsilon)^{p-1} - 1$ and $\theta = 2^p(1 + 1/\varepsilon)^{p-1}$, we get

$$x_u'^p \le (x_u + 2x_u^*)^p \le (1+\delta)x_u^p + \theta x_u^{*p}.$$

Thus,

$$x_u'^p - x_u^p \le \delta x_u^p + \theta x_u^{*p}.$$

Finally, if $u \notin V_a \cup V_b^*$, then $x'_u \le x_u$. We have,

$$\begin{aligned} \mathrm{cost}_{ab}(w) - \mathrm{cost}(w) &\le \sum_{u \in V_b^*} w_u \left(x_u'^p - x_u^p\right) + \sum_{u \in V_a \setminus V_b^*} w_u \left(x_u'^p - x_u^p\right) && (10) \\ &\le \sum_{u \in V_b^*} w_u \left(x_u^{*p} - x_u^p\right) + \sum_{u \in V_a} w_u \left(\delta x_u^p + \theta x_u^{*p}\right). && (11) \end{aligned}$$

We now derive an upper bound for $\xi_{ab} \cdot (\mathrm{cost}_{ab}(w) - \mathrm{cost}(w))$. Since $0 < m \le \xi_{ab} \le M$, we can replace ξ_{ab} with M in positive terms and with m in negative terms. We have:

$$\xi_{ab} \cdot (\mathrm{cost}_{ab}(w) - \mathrm{cost}(w)) \le \sum_{u \in V_b^*} w_u \left(M x_u^{*p} - m x_u^p\right) + M \sum_{u \in V_a} w_u \left(\delta x_u^p + \theta x_u^{*p}\right)$$

Now we assume that the swap $c_a \to c_b^*$ is sampled from the distribution $\mathcal{D}$ defined above. We get

$$\begin{aligned}&\mathbb{E}\left[\xi_{ab} \cdot (\mathrm{cost}_{ab}(w) - \mathrm{cost}(w))\right]\\ &\quad\le \sum_{u\in X} \Pr\{u \in V_b^*\} \cdot w_u \left(M x_u^{*p} - m x_u^p\right) + M \sum_{u\in X} \Pr\{u \in V_a\} \cdot w_u \left(\delta x_u^p + \theta x_u^{*p}\right)\end{aligned}$$

From 8, we get $\Pr\{u \in V_b^*\} = \frac{1}{k}$ and $\Pr\{u \in V_a\} \le \frac{2}{k}$ and thus

$$k\mathbb{E}\left[\xi_{ab} \cdot (\mathrm{cost}_{ab}(w) - \mathrm{cost}(w))\right] \le (1+2\theta)M\,\mathrm{cost}^*(w) - (m - 2\delta M)\,\mathrm{cost}(w). \tag{12}$$

We will use (12) to derive (4), (5), and (6). We will use different values of parameters ε, δ, and θ in these derivations. We will need the following lemma.

▶ **Lemma 13.** *For every $p \ge 1$ and every value of $\delta_0 \in (0, 1/4)$, there exists $\varepsilon > 0$ such that $\delta = (1+\varepsilon)^{p-1} - 1 \le \delta_0$ and $\theta \le 2^p \left(\frac{5p}{4\delta_0}\right)^{p-1}$.*

Proof. If $p = 1$, we let $\varepsilon = 1$, resulting in $\delta = 0$ and $\theta = 2$, and we are done. If $p > 1$, we choose ε so that $\delta = \delta_0$. Applying the Mean Value Theorem, we get

$$\varepsilon = (1+\delta_0)^{1/(p-1)} - 1 \ge \delta_0 \cdot \min_{t\in(0,\delta_0)} \frac{(1+t)^{1/(p-1)-1}}{p-1} \ge \frac{4}{5} \cdot \frac{\delta_0}{p-1},$$

where we used that $1 + t \le 1 + \delta_0 \le 5/4$ and $1/(p-1) - 1 \ge -1$. Therefore,

$$\theta = 2^p(1+1/\varepsilon)^{p-1} \le 2^p \left(\frac{5(p-1)+4\delta_0}{4\delta_0}\right)^{p-1} \le 2^p \left(\frac{5p}{4\delta_0}\right)^{p-1}. \qquad \blacktriangleleft$$

First, we establish the upper bound (4). We choose parameters according to Lemma 13 with $\delta_0 = m/(4M)$. Then $\theta \le 2^p(5pM/m)^{p-1}$. Since $0.5 \cdot 2^p(5Mp/m)^{p-1} \ge 2^{p-1} \ge 1$, we have

$$(1+2\theta)M \le 2.5 \cdot 2^p \left(\frac{5Mp}{m}\right)^{p-1} \cdot \frac{M}{m} \cdot m \le \frac{1}{2}\left(\frac{10Mp}{m}\right)^p m.$$

We conclude that

$$\mathbb{E}\left[\xi_{ab} \cdot (\mathrm{cost}_{ab}(w) - \mathrm{cost}(w))\right] \le \frac{m}{2k}\left(\left(\frac{10Mp}{m}\right)^p \mathrm{cost}^*(w) - \mathrm{cost}(w)\right).$$

Now we prove (5). We set $\varepsilon = 1$; then $\theta = 2^{2p-1}$ and $\delta = 2^{p-1} - 1$. From (12), we get

$$\begin{aligned}&k\mathbb{E}\left[\xi_{ab} \cdot (\mathrm{cost}_{ab}(w) - \mathrm{cost}(w))\right]\\ &\le (1+2\theta)M\,\mathrm{cost}^*(w) + 2\delta M\,\mathrm{cost}(w) \le (1+2^{2p})M(\mathrm{cost}(w) + \mathrm{cost}^*(w))\end{aligned}$$

Finally, for every swap $c_a \to c_b^*$ in the support of $\mathcal{D}$, we get the following upper bound from (11),

$$\begin{aligned}\mathrm{cost}_{ab}(w) - \mathrm{cost}(w) &\le \sum_{u\in V_b^*} w_u \left(x_u^{*p} - x_u^p\right) + \sum_{u\in V_a} w_u \left(\delta x_u^p + \theta x_u^{*p}\right)\\ &\le \sum_{u\in X} w_u x_u^{*p} + \sum_{u\in X} w_u \left(\delta x_u^p + \theta x_u^{*p}\right) \le (1+\theta)\,\mathrm{cost}^*(w) + \delta\,\mathrm{cost}(w).\end{aligned}$$

Choosing ε according to Lemma 13, we get the desired bound. $\blacktriangleleft$

5.5 Proof of Inequality 7

Applying Lemma 12, we get for $\alpha = 10Mp/m$,

$$\Delta_i \le \frac{m}{2k}\left(\alpha^p z_i^* - z_i\right) \tag{13}$$

$$\Delta_i \le \frac{(1+2^{2p})M}{k}(z_i + z_i^*). \tag{14}$$

From Lemma 12, item 3, with $\delta_0 = 1/(5r)$ and

$$\theta_0 = 2^p \left(\frac{25pr}{4}\right)^{p-1},$$

we get

$$\widehat{z}_i \le (1+\theta_0)z_i^* + (1+\delta_0)z_i \le \left(1+\frac{1}{5r}\right) z_i + (1+\theta_0)z_i^*. \tag{15}$$

Fix a threshold $T = \max\left(\frac{5r(1+\theta_0)}{4},\ 2(10eq)^p\right) = O(q)^p$. Consider two cases.

Current cost z_i is small: $z_i \le Tz_i^*$

From (15) and $1+\theta_0 \le \frac{4}{5r}T$,

$$\hat{z}_i \le \frac{6}{5}Tz_i^* + \frac{4}{5}Tz_i^* = 2Tz_i^*.$$

Accordingly, when $\hat{z}_i > z_i$, $M \le \varphi'(\hat{z}_i) = r\hat{z}_i^{r-1} \le r(2Tz_i^*)^{r-1}$. Also, $m \le M$. By (14)

$$\begin{aligned}\Delta_i &\le \frac{(4^p+1)M}{k}(z_i^* + z_i) = \frac{(4^p+1)}{k}\left(M\cdot(z_i^* + z_i + \frac{m}{M}z_i) - m\cdot z_i\right)\\ &\le \frac{(4^p+1)}{k}((2T+1)Mz_i^* - mz_i) \le \frac{1}{2k}\left(4^{p+2}\cdot r\cdot(2T)^r\varphi(z_i^*) - \varphi(z_i)\right).\end{aligned}$$

Letting $\beta = (2\cdot 4^{p+2}\cdot r\cdot(2T)^r)^{1/q} = O(q)$, we get

$$\Delta_i \le \frac{1}{4k}\left(\beta^q\varphi(z_i^*) - \varphi(z_i)\right).$$

Current cost z_i is large: $z_i \ge Tz_i^*$

We show that $2\alpha^p \le T$. From (15) and $1+\theta_0 \le \frac{4}{5r}T$, we have

$$\hat{z}_i \le \left(1 + \frac{1}{5r} + \frac{1+\theta_0}{T}\right) z_i \le \left(1+\frac{1}{r}\right) z_i.$$

Hence, when $\hat{z}_i > z_i$, $M \le \varphi'(\hat{z}_i) = r\hat{z}_i^{r-1} \le erz_i^{r-1}$ and $\alpha \le 10p\cdot(M/m) \le 10p\cdot(er) = 10eq$. Using that $z_i \ge Tz_i^* \ge 2(10eq)^p z_i^* \ge 2\alpha^p z_i^*$ and $\varphi(z_i) = mz_i$, we get from (13),

$$\Delta_i \le \frac{m}{2k}(\alpha^p z_i^* - z_i) \le -\frac{m}{2k}\cdot\frac{z_i}{2} = -\frac{1}{4k}\varphi(z_i).$$

6 Socially fair facility location

In this section, we consider two variants of the Socially Fair Facility Location problem and provide approximation algorithms for them.

We are given a set of n points X and a metric d on it, forming a metric space (X, d), as well as a set of possible facility locations F. There are ℓ groups on X represented by a weight function $w : X \to \mathbb{R}^{\ell}_{\geq 0}$ where $w_i(u) \geq 0$ denotes the weight of client u in group i. Each facility $c \in F$ is associated with a facility opening cost $f(c)$. We consider the following two settings with different types of facility opening costs: (1) group-homogeneous facility cost; (2) group-heterogeneous facility cost.

Group-homogeneous facility costs

Each facility $c \in F$ has a scalar opening cost $f(c) \geq 0$, paid equally across groups. The objective is to choose $C \subseteq F$ minimizing

$$\mathrm{FL}_{\mathrm{hom}}(w, f, C) := \left(\sum_{i=1}^{\ell} \mathrm{cost}(w_i, C)^{q/p} \right)^{1/q} + \sum_{c \in C} f(c),$$

where $\mathrm{cost}(w_i, C) = \sum_{u \in X} w_i(u)\, d(u, C)^p$.

[2] studied this objective with $p = 1$ and $q = \infty$ and gave a 4-approximation algorithm for it.

Group-heterogeneous facility costs

Each facility $c \in F$ has a vector opening cost $f(c) \in \mathbb{R}^{\ell}_{\geq 0}$, where group i pays $f_i(c)$ to open c.[5]

In this case, the objective is[6]

$$\mathrm{FL}_{\mathrm{het}}(w, f, C) := \left(\sum_{i=1}^{\ell} \left(\mathrm{cost}(w_i, C)^{1/p} + \sum_{c \in C} f_i(c) \right)^q \right)^{1/q}.$$

We get the following results for homogeneous and heterogeneous facility costs.

▶ **Theorem 5.** *For Socially Fair Facility Location with group-homogeneous facility cost, there exists a polynomial-time algorithm that achieves a $(4 + \varepsilon)$-approximation for any $\varepsilon > 0$.*

▶ **Theorem 6.** *For Socially Fair Facility Location with group-heterogeneous facility cost, there exists a polynomial-time algorithm that achieves an $O(q/\log q)$-approximation.*

We now provide the approximation algorithm for the group-heterogeneous setting.

---

[5] We interpret $f_i(c)$ as a group-specific *fixed* charge for opening facility c (e.g., taxes/subsidies or administrative overhead). Thus, group i pays $f_i(c)$ whenever c is opened, regardless of whether any group-i client is assigned to c; this keeps opening costs dependent only on the opened set C, as in standard fixed-charge facility location.

[6] If the heterogeneous opening costs are *group-uniform*, i.e., $f_i(c) = f(c)$ for all $i \in [\ell]$ and $c \in F$, then the objective becomes $\left(\sum_{i=1}^{\ell} \left(\mathrm{cost}(C, w_i)^{1/p} + \sum_{c \in C} f(c) \right)^q \right)^{1/q}$. We refer to this as the *group-uniform cost* variant. This objective is generally *different* from the group-homogeneous facility costs above. The two coincide when $q = \infty$, but they differ for finite q. By using our heterogeneous convex relaxation with $f_i(c) = f(c)$ and applying the rounding scheme used for the homogeneous costs, we obtain an $(8 + \varepsilon)$-approximation for this group-uniform cost variant.

Proof of Theorem 6. We use a relaxation of the problem and then use a randomized rounding procedure to get an integer solution. We consider the following relaxation

$$
\begin{aligned}
\min \quad & H \\
\text{s.t.} \quad & \sum_{i=1}^{\ell} z_i \le H^q \\
& z_i \ge \left(\sum_u w_i(u) \sum_{c \in F} d(u,c)^p \cdot x_{u,c} \right)^{\frac{q}{p}} && \forall i \in [\ell] && (16) \\
& z_i \ge \left(\sum_{c \in F} y_c \cdot f_i(c) \right)^q && \forall i \in [\ell] && (17) \\
& z_i \ge \sum_{c \in F} y_c \cdot f_i(c)^q && \forall i \in [\ell] && (18) \\
& \sum_{c \in F} x_{u,c} \ge 1 && \forall u \in X && (19) \\
& x_{u,c} \le y_c && \forall u \in X,\ \forall c \in F \\
& 0 \le y_c \le 1 && \forall c \in F \\
& 0 \le x_{u,c} \le 1 && \forall u \in X,\ \forall c \in F
\end{aligned}
$$

We can binary search on H. For any fixed value of H, the relaxation above is convex.

We then apply the following randomized rounding algorithm inspired by [19, 24]. Let (x^*, y^*, z^*, H^*) denote the fractional solution to the convex program above. The key difference is that, instead of opening the cheapest facility in B_u, we open exactly one facility c_u in B_u selected randomly with probability proportional to y^*_c. For each point $u \in X$, let $A_u = \sum_{c \in F} d(u,c)^p x^*_{u,c}$ denote the connection cost of point u given by the fractional solution. Then, we sort all points in X by their connection costs in increasing order. We iteratively perform the following steps until all points are assigned to open facilities. We first pick the unassigned point u with the smallest connection cost A_u as an anchor point. Let $B_u = \{c \in F : d(u,c)^p \le 2A_u\}$ be all facilities in a ball with radius $(2A_u)^{1/p}$ centered at u. It can be shown that the balls corresponding to distinct anchor points are disjoint. Then, we sample exactly one facility c_u in B_u with probability proportional to y^*_c. Assign all unassigned points v with distance $d(v,u) \le (2A_u)^{1/p} + (2A_v)^{1/p}$ to this new facility c_u. Let $\widehat{C}$ be the set of centers sampled by the rounding algorithm.

We now analyze the cost of this integer solution given by the rounding algorithm. Let $\hat{y}_c \in \{0,1\}$ be the indicator variable to denote whether facility c is selected in $\widehat{C}$. We first analyze the facility opening cost of this solution. By a simple averaging argument, for each iteration with anchor node u, we have $\sum_{c \in B_u} y^*_c \ge 1/2$. Since we open exactly one facility c_u in B_u with probability proportional to y^*_c, we have, for each facility $c \in B_u$,

$$\mathbb{E}[\hat{y}_c] = \frac{y^*_c}{\sum_{c \in B_u} y^*_c} \le 2y^*_c.$$

Consider any fixed group i. For each iteration with anchor point u, we define a random variable $Y_u = \sum_{c \in B_u} \hat{y}_c f_i(c)$ which represents the contribution to the facility cost of group i from that iteration. Since the choices made in distinct, disjoint anchor balls are independent, the collection $\{Y_u\}$ forms a set of independent non-negative random variables. By the

Rosenthal inequality (see [18]), we have

$$\mathbb{E}\left[\left(\sum_u Y_u\right)^q\right]^{1/q} \leq O\left(\frac{q}{\log q}\right)\max\left\{\left(\sum_u \mathbb{E}[Y_u^q]\right)^{1/q}, \sum_u \mathbb{E}[Y_u]\right\}.$$

Note that in each iteration, exactly one facility c_u in B_u is opened, which means exactly one random variable $\hat{y}_{c_u} = 1$ and all other $\hat{y}_c = 0$ in B_u. Thus, we have

$$\mathbb{E}[Y_u^q] = \mathbb{E}\left[\left(\sum_{c\in B_u} \hat{y}_c \cdot f_i(c)\right)^q\right] = \mathbb{E}\left[\sum_{c\in B_u} \hat{y}_c \cdot f_i(c)^q\right] \leq 2\sum_{c\in B_u} y_c^* \cdot f_i(c)^q.$$

Hence, we have $\sum_u \mathbb{E}[Y_u^q] \leq 2\sum_{c\in\bigcup_u B_u} y_c^* \cdot f_i(c)^q \leq 2z_i^*$. We also have

$$\sum_u \mathbb{E}[Y_u] = \sum_{c\in\bigcup_u B_u} f_i(c)\mathbb{E}[\hat{y}_c] \leq 2\sum_{c\in\bigcup_u B_u} f_i(c)y_c^* \leq 2(z_i^*)^{1/q} \leq O\left(\frac{q}{\log q}\right)H^*.$$

Therefore, we have

$$\mathbb{E}\left[\left(\sum_{c\in\bigcup_u B_u} \hat{y}_c f_i(c)\right)^q\right] \leq O\left(\left(\frac{2q}{\log q}\right)^q\right) z_i^*.$$

By taking the sum over all groups and applying Markov's inequality, with probability at least $1/2$, we have

$$\sum_{i=1}^{\ell}\left(\sum_{c\in\bigcup_u B_u} \hat{y}_c f_i(c)\right)^q \leq O\left(\left(\frac{2q}{\log q}\right)^q\right)\sum_{i=1}^{\ell} z_i^* \leq O\left(\left(\frac{2q}{\log q}\right)^q\right)(H^*)^q$$

We then analyze the connection cost of this integer solution. If v is assigned when u is the anchor, then $A_u \leq A_v$ and $d(v, c_u) \leq d(v, u) + d(u, c_u) \leq 3(2A_v)^{1/p}$. Hence, the connection cost for each group i given by $\widehat{C}$ is at most

$$\mathrm{cost}(w_i, \widehat{C})^{1/p} = \left(\sum_{u\in X} w_i(u)\cdot d(u,\widehat{C})^p\right)^{1/p} \leq \left(3^p\cdot 2\sum_{u\in X} w_i(u)\cdot A_u\right)^{1/p} \leq 6(z_i^*)^{1/q}.$$

By combining with the facility opening cost, the facility location cost is at most

$$\left(\sum_{i=1}^{\ell}\left(\mathrm{cost}(w_i,\widehat{C})^{1/p} + \sum_{c\in\widehat{C}} f_i(c)\right)^q\right)^{1/q}$$
$$\leq \left(\sum_{i=1}^{\ell} 2^q\,\mathrm{cost}(w_i,\widehat{C})^{q/p} + 2^q\left(\sum_{c\in\widehat{C}} f_i(c)\right)^q\right)^{1/q} \leq O\left(\frac{q}{\log q}\right)\cdot H^*.$$

Therefore, by computing a $(1+\varepsilon)$-approximate solution to the convex relaxation, this randomized rounding provides the desired approximation with probability at least $1/2$. ◀

References

1 Fateme Abbasi, Sandip Banerjee, Jarosław Byrka, Parinya Chalermsook, Ameet Gadekar, Kamyar Khodamoradi, Dániel Marx, Roohani Sharma, and Joachim Spoerhase. Parameterized approximation schemes for clustering with general norm objectives. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 1377–1399, 2023. doi:10.1109/FOCS57990.2023.00085.

2 Mohsen Abbasi, Aditya Bhaskara, and Suresh Venkatasubramanian. Fair clustering via equitable group representations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 504–514, 2021. doi:10.1145/3442188.3445913.

3 Sara Ahmadian, Alessandro Epasto, Marina Knittel, Ravi Kumar, Mohammad Mahdian, Benjamin Moseley, Philip Pham, Sergei Vassilvitskii, and Yuyan Wang. Fair hierarchical clustering. *Advances in Neural Information Processing Systems*, 33:21050–21060, 2020.

4 Sara Ahmadian, Alessandro Epasto, Ravi Kumar, and Mohammad Mahdian. Clustering without over-representation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 267–275, 2019. doi:10.1145/3292500.3330987.

5 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *Journal of the ACM (JACM)*, 42(4):844–856, 1995. doi:10.1145/210332.210337.

6 Barbara Anthony, Vineet Goyal, Anupam Gupta, and Viswanath Nagarajan. A plant location guide for the unsure: Approximation algorithms for min-max location problems. *Mathematics of Operations Research*, 35(1):79–101, 2010. doi:10.1287/MOOR.1090.0428.

7 Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing*, 33(3):544–562, 2004. doi:10.1137/S0097539702416402.

8 Sayan Bandyapadhyay, Tianzhi Chen, Zachary Friggstad, and Mahya Jamshidian. A constant-factor approximation for pairwise fair k-center clustering. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 43–57. Springer, 2025. doi:10.1007/978-3-031-93112-3_4.

9 Sayan Bandyapadhyay, Eden Chlamtáč, Zachary Friggstad, Mahya Jamshidian, Yury Makarychev, and Ali Vakilian. A polynomial-time approximation for pairwise fair k-median clustering. In *The 29th International Conference on Artificial Intelligence and Statistics*, 2026.

10 Sayan Bhattacharya, Parinya Chalermsook, Kurt Mehlhorn, and Adrian Neumann. New approximability results for the robust k-median problem. In *Scandinavian Workshop on Algorithm Theory*, pages 50–61. Springer, 2014. doi:10.1007/978-3-319-08404-6_5.

11 Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. Fair clustering through fairlets. *Advances in neural information processing systems*, 30, 2017.

12 Rajesh Chitnis, Marek Cygan, MohammadTaghi Hajiaghayi, Marcin Pilipczuk, and Michał Pilipczuk. Designing fpt algorithms for cut problems using randomized contractions. *SIAM Journal on Computing*, 45(4):1171–1229, 2016. doi:10.1137/15M1032077.

13 Eden Chlamtáč, Yury Makarychev, and Ali Vakilian. Approximating fair clustering with cascaded norm objectives. In *Proceedings of the Symposium on Discrete Algorithms (SODA)*, pages 2664–2683, 2022. doi:10.1137/1.9781611977073.104.

14 Mehrdad Ghadiri, Samira Samadi, and Santosh Vempala. Socially fair k-means clustering. In *Proceedings of the ACM Conference on Fairness, Accountability, and Transparency*, pages 438–448, 2021. doi:10.1145/3442188.3445906.

15 Mehrdad Ghadiri, Mohit Singh, and Santosh S Vempala. Constant-factor approximation algorithms for socially fair k-clustering. *arXiv preprint arXiv:2206.11210*, 2022. doi:10.48550/arXiv.2206.11210.

16 Dishant Goyal and Ragesh Jaiswal. Tight fpt approximation for socially fair clustering. *Information Processing Letters*, 182:106383, 2023. doi:10.1016/J.IPL.2023.106383.

17 Anupam Gupta and Kanat Tangwongsan. Simpler analyses of local search algorithms for facility location. *arXiv preprint arXiv:0809.2554*, 2008. arXiv:0809.2554.

18 William B Johnson, Gideon Schechtman, and Joel Zinn. Best constants in moment inequalities for linear combinations of independent and exchangeable random variables. *The Annals of Probability*, pages 234–253, 1985.

19 Adam Kasperski, Adam Kurpisz, and Paweł Zieliński. Approximability of the robust representatives selection problem. *Operations Research Letters*, 43(1):16–19, 2015. doi:10.1016/J.ORL.2014.10.007.

20 Sepideh Mahabadi and Ali Vakilian. Individual fairness for k-clustering. In *International conference on machine learning*, pages 6586–6596. PMLR, 2020. URL: http://proceedings.mlr.press/v119/mahabadi20a.html.

21 Konstantin Makarychev, Yury Makarychev, and Ilya Razenshteyn. Performance of johnson–lindenstrauss transform for k-means and k-medians clustering. *SIAM Journal on Computing*, 52(2):269–297, 2022.

22 Yury Makarychev and Ali Vakilian. Approximation algorithms for socially fair clustering. In *Conference on Learning Theory*, pages 3246–3264. PMLR, 2021. URL: http://proceedings.mlr.press/v134/makarychev21a.html.

23 David B Shmoys, Éva Tardos, and Karen Aardal. Approximation algorithms for facility location problems. In *Proceedings of the Symposium on Theory of Computing*, pages 265–274, 1997.

24 Aravind Srinivasan. Approximation algorithms via randomized rounding: a survey. *Series in Advanced Topics in Mathematics, Polish Scientific Publishers PWN*, pages 9–71, 1999.

Near-Optimal Streaming Approximation for Max-DICUT in Sublinear Space Using Two Passes

Santhoshini Velusamy
David R. Cheriton School of Computer Science, University of Waterloo, Canada

Abstract

The Max-DICUT problem has emerged as a canonical problem for understanding the approximability of constraint satisfaction problems in the streaming model. A seminal result of Kapralov and Krachun [STOC 2019] shows that it is impossible to beat 1/2-approximation for Max-DICUT in sublinear space in the single-pass streaming setting, even on bounded-degree graphs. In a recent work, Saxena, Singer, Sudan, and Velusamy [SODA 2025] prove that the above lower bound is tight by giving a single-pass algorithm for bounded-degree graphs that achieves $(1/2 - \epsilon)$-approximation in sublinear space, for every constant $\epsilon > 0$. For arbitrary graphs of unbounded degree, they give an $O(1/\epsilon)$-pass $O(\log n)$ space algorithm. Their work left open the question of obtaining 1/2-approximation for arbitrary graphs in the single-pass setting in sublinear space. We make progress towards this question and give a two-pass algorithm that achieves $(1/2 - \epsilon)$-approximation in sublinear space, for every constant $\epsilon > 0$.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Streaming, sublinear and near linear time algorithms; Theory of computation → Streaming models

Keywords and phrases Streaming algorithms, Approximation algorithms, Graph algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.4

Category APPROX

Related Version *Full Version*: `https://arxiv.org/abs/2512.19521`

Funding *Santhoshini Velusamy*: Supported in part by NSF award CCF-2348475 when the author was affiliated with Toyota Technological Institute at Chicago.

Acknowledgements Part of the work was conducted when the author was visiting the Simons Institute for the Theory of Computing as a research fellow in the Sublinear Algorithms program. The author thanks Raghuvansh Saxena, Noah Singer, and Madhu Sudan for collaborating on earlier investigations into this problem. The author thanks Aditya Bhaskara for discussions related to [5]. The author also thanks Sergio Thoumi for pointing out typos in a previous version.

1 Introduction

The maximum directed cut problem (Max-DICUT) is the analogue of the maximum cut (Max-CUT) problem for directed graphs. In particular, given a directed graph $G(V, E)$, the Max-DICUT problem asks to find an ordered bipartition (a "left" set L and a "right" set R) of the vertex set V that maximizes the number of edges that go from L to R. In contrast, in Max-CUT, we measure all the edges that cross L and R. Both Max-CUT and Max-DICUT are NP Hard and so, we study algorithms that approximately solve these problems. An α-approximation algorithm is one that outputs a solution that has a value which is at least α fraction of the optimum. Besides being fundamental problems in their own right, Max-CUT and Max-DICUT also belong to a large class of combinatorial optimization problems called the constraint satisfaction problems (CSPs) which capture several fundamental optimization problems in computer science. CSPs are well studied in theoretical computer science, owing in


© Santhoshini Velusamy;
licensed under Creative Commons License CC-BY 4.0

part to their structure which allows for finite characterizations of the infinite set of problems they contain, in several computational settings. Their study has led to the discovery of several important algorithmic and complexity-theoretic tools including SDP approximations, the PCP theorem, and Unique Games.

In many computational settings, Max-CUT has successfully served as a canonical problem in the study of CSPs. For example, in the polynomial time setting, Goemans and Williamson [14] gave the first SDP-based approximation algorithm for Max-CUT. And on the lower bound side, conditioned on the Unique Games conjecture, Khot, Kindler, Mossell, and O'Donnell [20] constructed tight integrality gap instances showing that the Goemans-Williamson SDP for Max-CUT is optimal. In a breakthrough follow-up work, Raghavendra [22] generalized both these works to all CSPs and gives an optimal SDP-based characterization for the approximability of every CSP in polynomial time. However, in the streaming setting, Max-CUT does not capture all CSPs.

In the single-pass streaming model of computation, the constraints of a CSP instance (or edges of a graph in the case of Max-CUT, Max-DICUT) arrive in a stream. The streaming algorithm is allowed one pass over this stream and is required to compute an approximation to the optimum *value* of the instance, using limited space. The amount of space available to the algorithm is typically parameterized by n, the number of variables in the instance (or vertices of the input graph in the case of Max-CUT, Max-DICUT). For every CSP, there exists a trivial sampling-based algorithm [1, 28] that can achieve $(1-\epsilon)$-approximation using $\tilde{O}(n)$ space. Therefore, the interesting regime is when the algorithm is allowed only $o(n)$ space. Surprisingly, Max-CUT has no non-trivial approximation in this regime! In particular, 1/2-approximation is trivial (just output 1/2 for every input graph) since any graph has a bipartition with at least half the edges crossing the bipartition. A series of works [17, 18, 19], culminating with the breakthrough result of Kapralov and Krachun [19] show that for any constant $\epsilon > 0$, any single-pass streaming algorithm that achieves $(1/2+\epsilon)$-approximation for Max-CUT requires $\Omega(n)$ space. A follow-up work due to Chou, Golovnev, Sudan, Velingker, and Velusamy [7] proves optimal inapproximability results for multiple CSPs in $o(n)$ space, generalizing techniques from the Max-CUT lower bound [19].

On the algorithmic side, the first non-trivial approximation algorithm was discovered by Guruswami, Velingker, and Velusamy [16] for Max-DICUT. While Max-DICUT has a trivial 1/4-approximation, they gave a single-pass algorithm that achieves $(2/5-\epsilon)$-approximation using only $O(\log n)$ space, for every constant $\epsilon > 0$. Chou, Golovnev, and Velusamy [10] later achieved $(4/9-\epsilon)$-approximation in $O(\log n)$ space. They also prove that the 4/9 factor is optimal in $o(\sqrt{n})$ space, generalizing the techniques from the $\Omega(\sqrt{n})$ space lower bound of Max-CUT due to Kapralov, Khanna, and Sudan [17]. They further generalized their results to prove a $O(\log n)$ vs $\Omega(\sqrt{n})$ space dichotomy theorem for every Max-2CSP. This served as the basis for their subsequent works with Sudan [8, 9] which prove a $O(\log^3 n)$ vs $\Omega(\sqrt{n})$ space dichotomy characterization for every CSP.[1] Saxena, Singer, Sudan, and Velusamy [24] show that the $\Omega(\sqrt{n})$ lower bound in [10] is tight and give a 0.485-approximation algorithm for Max-DICUT in $\tilde{O}(\sqrt{n})$ space. In a later work [25], the same authors also show that it is possible to achieve $(1/2-\epsilon)$-approximation for *bounded-degree* (i.e., maximum degree bounded by a constant) graphs in $O(n^{1-\Omega(1)})$ space, for every $\epsilon > 0$. A trivial reduction from Max-CUT shows that the factor 1/2 is also optimal for Max-DICUT in $o(n)$ space.

[1] Their dichotomy characterization is for *dynamic* streaming algorithms which allow deletions and can process arbitrarily long streams.

In the multi-pass setting, a streaming algorithm is allowed to do multiple passes over the same stream of constraints. Assadi and Nagargoje [3] show that any p-pass algorithm for Max-CUT achieving $(1-\epsilon)$-approximation requires at least $\Omega(n^{1-O(\epsilon p)})$ space. A recent breakthrough result of Fei, Minzer, and Wang [11] shows that for any $\epsilon > 0$, any p-pass algorithm for Max-CUT (also Max-DICUT) achieving $(1/2+\epsilon)$-approximation requires at least $\Omega(n^{1/3}/p)$ space. In contrast, for *dense* instances of average degree at least n^{δ}, a two-pass streaming algorithm due to Bhaskara, Daruki, and Venkatasubramanian [5], achieves $(1-\epsilon)$-approximation for Max-CUT (also Max-DICUT) using $\tilde{O}(n^{1-\delta})$ space. An algorithm due to Saxena, Singer, Sudan, and Velusamy achieves $(1/2-\epsilon)$-approximation on arbitrary graphs in $O(1/\epsilon)$ passes using $O(\log n)$ space [25]. In a prior work [23], the same authors give a two-pass algorithm that achieves 0.485-approximation using $O(\log n)$ space. A recent result due to Fei, Minzer, and Wang [12] gives a dichotomy characterization in the multi-pass setting based on the basic LP relaxation for every CSP. In particular, they show that the techniques from [27, 25] can be combined with the distributed algorithms of Yoshida [29] to obtain a streaming algorithm that solves the LP relaxation for every CSP in $O(1/\epsilon^2)$ passes and $O(\log n)$ space. And to beat the approximation achieved by the basic LP, any p-pass streaming algorithm requires at least $\Omega(n^{1/3}/p)$ space.

The connection between streaming algorithms and LP relaxations was also identified in a concurrent work by Singer, Tulsiani, and Velusamy [26]. They conjecture that for every CSP, there must be a single-pass $o(n)$ space streaming algorithm that solves its LP relaxation, and any algorithm beating this factor must require at least $\Omega(n)$ space. For Max-DICUT, its basic LP achieves 1/2-approximation and so their conjecture states that there is a single-pass algorithm that achieves 1/2-approximation in $o(n)$ space. In this work, we make progress towards this conjecture and prove it in the two-pass setting.

▶ **Theorem 1.1** (Main result). *For every $\epsilon > 0$, there is a two-pass streaming algorithm that achieves $(1/2-\epsilon)$-approximation for* Max-DICUT *on any n-vertex graph using space $O(n^{1-\Omega_\epsilon(1)})$.*

2 Technical overview

In this section, we present a technical overview of the proof of our main result.

There are three main ingredients to our proof: 1) a single-pass $(1/2-\epsilon)$-approximation algorithm of Saxena, Singer, Sudan, and Velusamy [25] for bounded-degree graphs that uses $O(n^{1-\Omega(1)})$ space, 2) Trevisan's reduction [27] from unbounded-degree to bounded-degree graphs that approximately preserves the Max-DICUT value, and 3) Bhaskara, Daruki, and Venkatasubramanian's [5] two-pass algorithm that achieves $(1-\epsilon)$-approximation on graphs of average degree at least n^{δ} using $\tilde{O}(n^{1-\delta})$ space. The following theorems formally state their results.

▶ **Theorem 2.1** ([25]). *For every $d \in \mathbb{N}$ and $\epsilon > 0$, there is a single-pass streaming algorithm that achieves $(1/2-\epsilon)$-approximation for* Max-DICUT *on any n-vertex graph of maximum degree at most d using space $O(n^{1-\Omega(1)})$.*

▶ **Theorem 2.2** ([27]). *For any directed graph $G(V,E)$ on n vertices and m edges, and $0 < \epsilon \leq 1$, there exists a (random) directed graph $\overline{G}(\overline{V},\overline{E})$ on $2m$ vertices and $O(m/\epsilon^2)$ edges such that with probability at least $5/6$, $\textsf{Max-DICUT}(G)-\epsilon' \leq \textsf{Max-DICUT}(\overline{G}) \leq \textsf{Max-DICUT}(G)+\epsilon$, and the maximum degree of a vertex in $\overline{G}$ is at most $O(1/\epsilon^2)$.*

▶ **Theorem 2.3** ([5]). *For every $\epsilon, \delta \in (0, 1]$, there is a two-pass streaming algorithm that achieves $(1-\epsilon)$-approximation for* Max-DICUT *on any n-vertex graph of average degree at least n^{δ}, using space $\tilde{O}(n^{1-\delta})$.*

At a high level, our idea is to reduce the input graph which could potentially have an unbounded maximum degree to a bounded-degree graph using Trevisan's reduction [27], and then execute the [25] algorithm for bounded-degree graphs. However, there are challenges to this naive approach. Firstly, Trevisan's reduction is not a small-space reduction and requires the knowledge of the degree of every vertex in the graph. This would be too expensive to store in the streaming setting. Secondly, the reduction produces a graph with $2m$ vertices, which could be $\Theta(n^2)$ in the worst case, hence the [25] algorithm could require $\Omega(n)$ space. We deal with the second challenge by running the [25] algorithm only when the input graph is sufficiently sparse, i.e., has at most $n^{1+\delta}$ edges, for some suitably chosen δ so that the space usage is still at most $O(n^{1-\Omega(1)})$. And when the input instance is dense, we run the [5] algorithm. Thus, the space usage is always at most $O(n^{1-\Omega(1)})$. To deal with the first challenge, we implement the [25] algorithm in a non black-box manner. We do not produce the entire bounded-degree graph from the Trevisan reduction and instead only sample parts of this graph that are required for the [25] algorithm. Below, we give a high-level overview of both these algorithms and our implementation of them.

Trevisan reduction. The reduction is quite simple. Say $G(V, E)$ is an unweighted directed graph on n vertices and m edges, with possible multi-edges but no self-loops. Given G, Trevisan first creates a weighted graph $G'(V', E')$: for every vertex $v \in V$, he adds $\deg(v)$ many "copies" of it, labeled $(v, 0), \ldots, (v, \deg(v) - 1)$, to V'. For every edge $u \to v \in E$, and $i \in [\deg(u)], j \in [\deg(v)]$,[2] he adds an edge $(u, i) \to (v, j)$ of weight $\frac{1}{\deg(u)\deg(v)}$ to E'. It is not hard to see that the Max-DICUT values of G and G' are equal.[3] He then samples a random graph $\tilde{G}$ based on G' in the following way. The vertex set of $\tilde{G}$ is V' and he samples the edge set as follows: he sets a parameter $d = O(1/\epsilon^2)$ and samples dm edges independently (with repetition) from G', where in each sample, an edge in E' is sampled with probability proportional to its weight. He finally deletes a minimal set of edges from $\tilde{G}$ so that the degree of every vertex is at most $O(d)$. He argues that with high probability, at most an ϵ fraction of the edges are deleted, and that the Max-DICUT value is preserved up to an additive ϵ factor.

SSSV algorithm. Saxena, Singer, Sudan, and Velusamy make a key observation that on bounded-degree graphs, in order to simulate a distributed algorithm for Max-DICUT, it suffices to estimate certain neighborhood distributions of the input graph (see Section 3.1 for more details). They show that such distributions can be estimated from the induced subgraph of a randomly sampled subset of the vertex set. Finally, based on a previous distributed algorithm of Censor-Hillel, Levy, and Shachnai [6], they give a distributed algorithm that achieves $(1/2 - \epsilon)$-approximation for Max-DICUT, and efficiently simulate it in sublinear space in the single-pass setting using the above techniques.

[2] For a natural number $n \in \mathbb{N}$, we define $[n] = \{0, \ldots, n-1\}$. Note that this is slightly different from the standard definition.

[3] For every $u \in V$ and $i, i' \in [\deg(u)]$, the copies (u, i) and (u, i') have identical neighborhood structure. Hence, for any fixed assignment of the remaining vertices, the DICUT value is locally maximized by placing all copies of u on the same side of the cut.

Putting them together (a three-pass algorithm). Since the [25] algorithm only requires the induced subgraph of a randomly sampled subset of the vertex set, we do not need to do the Trevisan reduction on the entire input graph. Instead, we directly sample a random subset of vertices from the bounded-degree graph, i.e., from $V' = \{(v, i) : v \in V; i \in [\deg(v)]\}$, and perform a *partial* Trevisan reduction on their *parent* vertices. To sample from V', we first map every edge $e \in E$, based on its position in the stream, to two unique vertices in V', i.e., no two edges map to the same vertex. In particular, an edge $u \to v$ is mapped to (u, i) and (v, j) iff it is the i-th edge (resp. j-th edge) incident on u (resp. v) in the stream. Thus, as the edges arrive in the stream, we can sample from V'.[4] Once the vertices are sampled, we have to sample the induced subgraph on these vertices. The original Trevisan reduction performs a *global* sampling of the edges, where every edge is sampled with probability proportional to its weight. In order to perform such a sampling, one would have to compute the weights on all the edges, which is infeasible in the streaming setting. We therefore modify the reduction to a *local* sampling procedure, where for every edge $u \to v \in E$, we create $d = O(1/\epsilon^2)$ "copies" of it in E' and for every copy of the edge, we independently sample a uniform random copy of u and a uniform random copy of v in V' as its endpoints.[5]We also delete *all* the edges incident on high-degree vertices and not just the minimal set of edges, as it is not clear how to compute the minimal set without generating the entire graph. We argue in Section 4 that this procedure still preserves the Max-DICUT value up to an additive ϵ factor with high probability. Now that we have this local procedure, we generate the induced subgraph on the sampled vertices as follows. After we sample the vertices in the first pass, we store the degrees of their parent vertices in the second pass. In the third pass, we again scan through the edges in the stream and if $u \to v$ is such that both u and v have at least one of their copies sampled, then we sample d independent copies of the edge as outlined above (we can do this since we know the degrees of u and v), and we store those edges that are in the induced subgraph. If neither u nor v have any copies in the sampled set of vertices, then we do nothing. If only one of them, say u, has sampled copies, then we sample only the endpoints corresponding to u's copies for the d copies of the edge. We do this to keep track of the degrees of all the sampled vertices, and in the end, delete all the edges incident on the high-degree vertices. The formal description of the procedure and the proof of correctness are described in Section 5.

Modifications to obtain a two-pass algorithm. In Section 6, we modify the above algorithm and get a two-pass algorithm by getting rid of the second pass which computes and stores the degrees of the parent vertices *exactly*. In particular, we show that our modified Trevisan reduction in Section 4 works even when we have only *approximate* estimates for the degrees of high-degree vertices. In the first pass, along with sampling vertices like in the above algorithm, we estimate the degrees of high-degree vertices using standard sampling methods. In particular, for every vertex, we sample each edge incident to it independently with probability $p = n^{-O_\epsilon(1)}$ and rescale the total number of sampled edges by $1/p$ to estimate its degree. This procedure would estimate the degrees accurately (up to a multiplicative error) for vertices of sufficiently-large polynomial degree, say n^c. We can ensure that the

[4] There is a subtlety here. We do not have enough space to remember the ordering of the edges incident on every vertex in the input graph, which is crucial in the above mapping. Instead, we label the sampled copies of a vertex based on the order in which they are sampled by our algorithm, and argue that this relabeling does not affect the correctness of the algorithm (see Section 5 for more details).

[5] A slightly different reduction, also amenable to streaming, was recently proposed by Fei, Minzer, and Wang in [12].

space is still sublinear because we run the Trevisan procedure only on graphs with bounded average degree. We then use these estimates to run edge sampling as in the third pass of the above algorithm. However, the copies of the edges incident on low-degree vertices will not be sampled correctly in this step. To fix this, for every vertex that is sampled in the first pass, we store the first n^c edges incident on its *parent* vertex in the second pass.[6] Thus, we can identify parent vertices of degree less than n^c, and also compute their degrees *exactly*. In post-processing, we delete the edges incident to the copies of such vertices, and resample them according to their correct distributions (since we now have the exact degrees of the low-degree parent vertices).

BDV algorithm for Max-CUT. Finally, we give a high-level description of the two-pass algorithm for dense instances of Max-CUT in [5]. While their paper explicitly states the result only for Max-CUT, their algorithm immediately extends to Max-DICUT as well. Their main idea is to use certain non-uniform sampling methods to construct a "core-set" of *sublinear* size for the input graph, which preserves the Max-CUT value up to an additive error ϵ. A core-set of linear size can be obtained for any graph by uniformly sampling $\tilde{O}(n/\epsilon^2)$ edges [1, 28]. And for very dense graphs of average degree at least $\Omega(n)$, constant-size core-sets can be constructed [15, 2, 21]. In the polynomial density regime (i.e., average degree n^δ for $\delta < 1$), previous works [13, 4] show that if both the maximum and the minimum degrees are $\Theta(n^\delta)$, then a core-set of size $\tilde{O}(n^{1-\delta})$ exists. Bhaskara, Daruki, and Venkatasubramanian [5] extend this result to general graphs in this regime, and give a two-pass streaming algorithm to construct a core-set of size $\tilde{O}(n^{1-\delta})$.

3 Preliminaries

In this paper, any reference to a directed graph G means unweighted graphs with multi-edges but no self-loops. For a directed graph $G(V, E)$, a dicut is an ordered bipartition of the vertex set V. An edge $u \to v$ is said to belong to a dicut $L \sqcup R$ if $u \in L$ and $v \in R$. The value of the dicut, denoted by $\mathsf{DICUT}(G, L \sqcup R)$ is the fraction of edges in G that belong to the dicut $L \sqcup R$. We define $\mathsf{Max\text{-}DICUT}(G)$ to be $\max_{L \sqcup R} \mathsf{DICUT}(G, L \sqcup R)$. For any function $\rho : V \to [0, 1]$, we define the expected dicut value of G under the rounding ρ, denoted by $\mathsf{DICUT}(G, \rho)$, to be $\mathbb{E}[\mathsf{DICUT}(G, L \sqcup R)]$, where every vertex in V belongs to L independently with probability $\rho(v)$.

For any function $f : A \to B$ that maps a domain A to a co-domain B, and $C \subseteq A$, we denote the restriction of the function to the domain C by $f|_C : C \to B$.

Distribution Notation

Let $\Delta(S)$ denote the set of all probability distributions over a finite set S. A distribution $\mu \in \Delta(S)$ assigns to each element $x \in S$ a probability $\mu(x) \geq 0$ such that $\sum_{x \in S} \mu(x) = 1$.

Total Variation Distance

The *total variation distance* between two distributions $\mu, \nu \in \Delta(S)$ is defined as

$$d_{\mathrm{TV}}(\mu, \nu) = \frac{1}{2} \sum_{x \in S} |\mu(x) - \nu(x)|.$$

[6] We need to be a bit careful when defining the degree thresholds and the sampling probabilities, but they can be set in a way to keep the space sublinear.

Equivalently,

$$d_{\mathrm{TV}}(\mu,\nu) = \max_{A\subseteq S} |\mu(A) - \nu(A)|,$$

where $\mu(A) = \sum_{x\in A}\mu(x)$. The total variation distance satisfies $d_{\mathrm{TV}}(\mu,\nu) \in [0,1]$, with $d_{\mathrm{TV}}(\mu,\nu) = 0$ if and only if $\mu = \nu$, and $d_{\mathrm{TV}}(\mu,\nu) = 1$ if and only if the supports of μ and ν are disjoint.

We now list some standard concentration bounds that will be used in the rest of the paper.

▶ **Theorem 3.1** (Chebyshev's inequality). *Let X be a random variable with variance σ^2. For any $a > 0$, we have*

$$\Pr[|X - \mathbb{E}[X]| > a] \leq \frac{\sigma^2}{a^2}.$$

▶ **Theorem 3.2** (Chernoff bound). *Let $X_1, \ldots, X_n$ be independent random variables taking values in $\{0,1\}$. Consider the sum of these random variables, $S_n = X_1 + \cdots + X_n$. Let $\mu = \mathbb{E}[S_n]$. Then for $R \geq 2e\mu$,*

$$\Pr[S_n \geq R] \leq 2^{-R}.$$

▶ **Theorem 3.3** (Hoeffding's inequality). *Let $X_1, \ldots, X_n$ be independent bounded random variables such that $a_i \leq X_i \leq b_i$. Consider the sum of these random variables, $S_n = X_1 + \cdots + X_n$. Then Hoeffding's theorem states that for all $t > 0$,*

$$\Pr[|S_n - \mathbb{E}[S_n]| \geq t] \leq 2\exp\left(-\frac{2t^2}{\sum_{i=1}^n (b_i - a_i)^2}\right).$$

3.1 Neighborhood distributions and the simulation of local algorithms

A key contribution of Saxena, Singer, Sudan, and Velusamy in [25] is the identification of a connection between streaming and local algorithms, where they show that it is possible to estimate local neighborhood distributions of graphs in the streaming setting and use these distributions to simulate constant-round distributed algorithms. We recall some of the notations introduced in their paper and reprove their main theorem that shows that sampling vertices uniformly at random and estimating local neighborhood distributions from the induced subgraph suffices to estimate the value of a local algorithm that is executed on the original graph.

First, we formally define the LOCAL model for distributed algorithms for the Max-DICUT problem and state the result of [6, 25].

▶ **Definition 3.4** (The LOCAL model for distributed algorithms for Max-DICUT). *Let $G(V,E)$ be a directed graph on n vertices and m edges. G describes the topology of a communication network, where each node can communicate with both its incoming and outgoing neighbors. Communication happens in synchronous rounds, where in every round, every node does some internal computations and sends an arbitrarily long message to each of its neighbors. At the end, each node v computes a fractional assignment $\rho(v) \in [0,1]$ and the approximation ratio achieved by the algorithm is defined to be $\mathsf{DICUT}(G,\rho)/\mathsf{Max\text{-}DICUT}(G)$. In randomized algorithms, the nodes also have access to an arbitrarily long private random bit string that is sampled prior to any communication between the nodes.*

▶ **Theorem 3.5** ([6, 25]). *For every $\epsilon > 0$, there exists a distributed algorithm for* Max-DICUT *in the LOCAL model that achieves $(1/2-\epsilon)$-approximation with probability at least $9/10$, and requires at most $1/\epsilon$ rounds of communication. It is a randomized algorithm where every node uses only an $O(\log(1/\epsilon))$-bit private random string, and the random strings can be sampled from a pairwise independent hash family.*

Saxena, Singer, Sudan, and Velusamy [25] observed that for each vertex v, $\rho(v)$ depends only on the structure of a ball of radius $1/\epsilon$ centered at that node, and the private random strings of all the nodes in this ball. They called this the "neighborhood type" of a vertex. They observed that in order to compute $\mathsf{DICUT}(G,\rho) = (\sum_{u\to v\in E}\rho(u)(1-\rho(v)))/m$ with a small additive error and a constant success probability, it suffices to sample a random constant-sized subset $S \subset E$ and output $\sum_{u\to v\in S}\rho(u)(1-\rho(v))/|S|$. This motivated them to define the notion of a neighborhood type of an edge that captures the neighborhood types of both its endpoints. Thus, estimating the distribution of the edge-neighborhood types would suffice to estimate $(\sum_{u\to v\in E}\rho(u)(1-\rho(v)))/m$, which in turn gives a $(1/2-\epsilon)$-approximation to $\mathsf{Max\text{-}DICUT}(G)$. For graphs whose maximum degree is bounded by a constant, the set of all possible edge-neighborhood types is constant sized, and hence, they could estimate their distribution efficiently in the streaming setting. In what follows, we will set up some notations to define the neighborhood types and reprove a key theorem from [25].

Let $G(V,E)$ be a directed graph and $u, v \in V$. We say that v is of distance at most ℓ from u if there exists a *path* from u to v which is a sequence of vertices $u = w_0, w_1, \ldots, w_{\ell-1}, w_\ell = v \in V$ such that for each $i \in [\ell]$, $w_{i-1} \to w_i \in E$ or $w_i \to w_{i-1} \in E$. Let $\mathbf{Typ}_c^{\ell,D}$ denote the set of all isomorphism classes of doubly rooted labeled graphs where every vertex is of degree at most D, has a c-bit label, and is of distance at most ℓ from (at least) one of the roots. Consider a directed graph $G(V,E)$ whose maximum degree is at most D. Let $\mathcal{H}_n$ be the pairwise independent hash family that is used to sample the private random strings for every node in the distributed algorithm of [6, 25]. Let $h : V \to \{0,1\}^c$ be any hash function in $\mathcal{H}_n$. For an edge $e \in E$, consider the following doubly rooted labeled graph $\mathcal{G}_e$: $\mathcal{G}_e$ is rooted at the endpoints of e and consists of all the vertices in G which are of distance at most ℓ from (at least) one of the roots, and their corresponding induced subgraph. In addition, each vertex v in G_e is labeled $h(v)$. The radius-ℓ neighborhood type of e with respect to G and the hash function h, $\mathsf{nbhdtype}^{\ell}_{G,h}(e)$, is defined to be the representative of the isomorphism class in $\mathbf{Typ}_c^{\ell,D}$ that contains $\mathcal{G}_e$. The neighborhood edge-type distribution of G with respect to the hash function h is defined as follows.

▶ **Definition 3.6** (Edge-type distribution). *Let $h : V \to \{0,1\}^c$, $\ell, D \in \mathbb{N}$ and $G = (V,E)$ be a directed graph of maximum degree D. The* radius-ℓ neighborhood type distribution *of G with respect to h, denoted $\mathsf{EdgeNbhdTypeDist}^{\ell}_{G;h}$, is the distribution over $\mathbf{Typ}_c^{\ell,D}$ given by sampling a random $e \sim \mathsf{Unif}(E)$ and outputting $\mathsf{nbhdtype}^{\ell}_{G,h}(e)$.*

▶ **Corollary 3.7** ([6, 25]). *For every $\epsilon'' > 0$, there exist $\ell, c, n_0 \in \mathbb{N}$ and a function $\mathtt{Local} : \mathbf{Typ}_c^{\ell,D} \to [0,1]$ such that for every $n \geq n_0$, there exists a hash family $\mathcal{H}_n$ such that the following holds. Let $G = (V,E)$ be a directed graph on n vertices and of maximum degree D. Let $h \sim \mathsf{Unif}(\mathcal{H}_n)$. Then with probability at least $9/10$,*

$$\frac{\mathsf{Max\text{-}DICUT}(G)}{2} - \epsilon'' \leq \mathbb{E}_{T\sim\mathsf{EdgeNbhdTypeDist}^{\ell}_{G;h}}[\mathtt{Local}(T)] \leq \mathsf{Max\text{-}DICUT}(G).$$

The following corollary implies that it suffices to compute a distribution that is close in total variation distance to $\mathsf{EdgeNbhdTypeDist}^{\ell}_{G;h}$.

▶ **Corollary 3.8.** *If $\mathcal{D} \in \Delta(\mathbf{Typ}_c^{\ell,D})$ is such that $\mathsf{tvd}(\mathcal{D}, \mathsf{EdgeNbhdTypeDist}^{\ell}_{G;h}) \leq \epsilon''$, then*

$$\frac{\mathsf{Max\text{-}DICUT}(G)}{2} - 2\epsilon'' \leq \mathbb{E}_{T\sim\mathcal{D}}[\mathtt{Local}(T)] \leq \mathsf{Max\text{-}DICUT}(G) + \epsilon''.$$

The following theorem (combined with Corollary 3.8) shows that for bounded-degree graphs, in order to achieve a $(1/2-\epsilon)$-approximation to Max-DICUT, it suffices to sample a sublinear number of random vertices and compute a "rescaled" edge-type neighborhood distribution of the induced subgraph.

▶ **Theorem 3.9** (Simplified non-streaming version of Algorithm 2 in [25]). *For every $\epsilon'' > 0, \alpha \in [0,1), D, c, \ell \in \mathbb{N}$, there exist $\beta_0 \in (0,1)$ and $n_0 \in \mathbb{N}$ such that for all $n \geq n_0$ and $n^{-\beta_0} \leq p \leq 1$, the following hold: Let $G(V,E)$ be a directed graph on n vertices and m edges such that the maximum degree is at most D and at most α fraction of the vertices are isolated. Let $h \in V \to \{0,1\}^c$. Let $V' \subset V$ be a random subset where every vertex in V is sampled independently with probability p. Let $G' \subseteq G$ be the induced subgraph of G on the vertices in V'. For every $T \in \mathbf{Typ}_c^{\ell,D}$, let a_T denote the number of edges e in G' such that all the vertices in G which are at a distance of at most ℓ from at least one of the endpoints of e, are contained in V', and $\mathsf{nbhdtype}^{\ell}_{G',h|_{V'}}(e) = T$. Consider the distribution $\mathcal{D} \in \Delta(\mathbf{Typ}_c^{\ell,D})$, defined to be*

$$\mathcal{D}(T) = \frac{a_T \cdot p^{-|T|}}{\sum_{T' \in \mathbf{Typ}_c^{\ell,D}} a_{T'} \cdot p^{-|T'|}},$$

where $T \in \mathbf{Typ}_c^{\ell,D}$ and $|T|$ denotes the number of vertices in T. Then, with probability at least $9/10$, $\mathsf{tvd}(\mathcal{D}, \mathsf{EdgeNbhdTypeDist}^{\ell}_{G;h}) \leq \epsilon''$.

A formal proof of this theorem appears in the full version.

4 Reduction from unbounded-degree graphs to bounded-degree graphs

In this section, we give a generalized version of the Trevisan reduction [27] that is tailored to be implementable in the two-pass streaming setting in Section 6.

Algorithm 1 Algorithm for reducing unbounded-degree graphs to bounded-degree graphs.

Parameters: $n, m \in \mathbb{N}$, $\epsilon' > 0$, $\zeta > 0$
Input:
A directed graph $G(V,E)$ on n vertices and m edges
An n-length array $\mathsf{approx\text{-}deg}$ (indexed by V) such that when $\deg(v) < n^{\zeta}$, we have $\mathsf{approx\text{-}deg}(v) = \deg(v)$ and when $\deg(v) \geq n^{\zeta}$, we have $(1-\epsilon'/100)\deg(v) \leq \mathsf{approx\text{-}deg}[v] \leq (1+\epsilon'/100)\deg(v)$

Initialize directed graphs $\tilde{G}(\tilde{V},\tilde{E}), \overline{G}(\overline{V},\overline{E})$, where $\tilde{V} = \overline{V} = \{(v,i) : v \in V, i \in [\deg(v)]\}$ and $\tilde{E} = \emptyset, \overline{E} = \emptyset$
Set a parameter $d = 80/(\epsilon')^2$
for every edge $u \to v \in E$ **do**
    Initialize $\mathsf{i} = 1$
    for $\mathsf{i} \leq d$ **do**
        Sample $i_1 \sim \mathsf{Unif}([\mathsf{approx\text{-}deg}(u)])$
        Sample $i_2 \sim \mathsf{Unif}([\mathsf{approx\text{-}deg}(v)])$
        if $i_1 < \deg(u)$ and $i_2 < \deg(v)$ **then**
            Add $(u,i_1) \to (v,i_2)$ to $\tilde{E}$
        Increment i by 1
Copy all the edges in $\tilde{E}$ to $\overline{E}$
for every $v \in \overline{V}$ such that $\deg(v) > 11d$ **do**
    Delete all the edges incident on v in $\overline{E}$
Output $\overline{G}(\overline{V},\overline{E})$

▶ **Theorem 4.1.** *Let $n \geq 2, m \geq 10 \in \mathbb{N}$. For any directed graph $G(V,E)$ on n vertices and m edges, and $0 < \epsilon' \leq 1$, $\zeta > 0$, with probability at least $5/6$, Algorithm 1 outputs a directed graph $\overline{G}(\overline{V},\overline{E})$ on $2m$ vertices and $\Theta(m/(\epsilon')^2)$ edges such that $\mathsf{Max\text{-}DICUT}(G) - \epsilon' \leq \mathsf{Max\text{-}DICUT}(\overline{G}) \leq \mathsf{Max\text{-}DICUT}(G) + \epsilon'$, and the maximum degree of a vertex in $\overline{G}$ is at most $O(1/(\epsilon')^2)$.*

Proof. Recall the construction of $\tilde{G}$ in Algorithm 1. We will first argue that the number of edges in $\tilde{E}$ is at least $md(1-\epsilon'/8)$, with high probability. Observe that for any edge $u \to v \in E$, when we sample its i-th copy in $\tilde{G}$, the probability that either i_1 or i_2 fail to satisfy the condition in Algorithm 1 is at most $\epsilon'/50$. Therefore, applying Chernoff bound (Theorem 3.2), we conclude that with probability at least $1-\exp(-10m)$, $|\tilde{E}| \geq md(1-\epsilon'/8)$.

Max-DICUT value of $\tilde{G}$. Fix any ordered bipartition $L \sqcup R$ of $\tilde{V}$. For an edge $e \in E$, let $X_{e,\mathsf{i}}$ be the indicator random variable for the event that in the i-th iteration corresponding to edge e, the condition in Algorithm 1 is satisfied and the corresponding edge in $\tilde{E}$ belongs to the dicut $L \sqcup R$. Then, the DICUT value of $\tilde{G}$ under $L \sqcup R$ is given by

$$\mathsf{DICUT}(\tilde{G}, L \sqcup R) = \frac{\sum_{e\in E, 1\leq \mathsf{i}\leq d} X_{e,\mathsf{i}}}{|\tilde{E}|}\,.$$

Let us define a function $\rho_{L\sqcup R} : V \to [0,1]$ such that for $v \in V$, $\rho_{L\sqcup R}(v) = (|L \cap \{(v,i) : i \in [\deg(v)]\}|)/\deg(v)$, i.e., $\rho_{L\sqcup R}(v)$ is the fraction of copies of v in L. Then,

$$\mathsf{DICUT}(G, \rho_{L\sqcup R}) - \epsilon'/33 \leq \frac{\sum_{e\in E, 1\leq \mathsf{i}\leq d} \mathbb{E}[X_{e,\mathsf{i}}]}{dm} \leq \mathsf{DICUT}(G, \rho_{L\sqcup R}) + \epsilon'/33\,.$$

This is because for every $u \to v \in E, 1 \leq \mathsf{i} \leq d$,

$$\rho_{L\sqcup R}(u)(1-\rho_{L\sqcup R}(v)) - \epsilon'/33 \leq X_{u\to v,\mathsf{i}} \leq \rho_{L\sqcup R}(u)(1-\rho_{L\sqcup R}(v)) + \epsilon'/33\,, \tag{4.1}$$

and

$$\mathsf{DICUT}(G,\rho) = \frac{\sum_{u\to v} \rho(u)(1-\rho(v))}{m}\,.$$

To see why Equation (4.1) is true, observe that for $i_1 \sim \mathsf{Unif}([\mathsf{approx\text{-}deg}[u]])$, we have

$$\rho_{L\sqcup R}(u) - \epsilon'/99 \leq \Pr[i_1 < \deg(u) \text{ and } (u,i_1) \in L] \leq \rho_{L\sqcup R}(u) + \epsilon'/99\,,$$

since for every $u \in V$, $\mathsf{approx\text{-}deg}[u]$ satisfies

$$(1-\epsilon'/100)\deg(u) \leq \mathsf{approx\text{-}deg}[u] \leq (1+\epsilon'/100)\deg(u)\,,$$

Since all the edges were sampled independently and $d = 80/(\epsilon')^2$, we can apply Hoeffding's bound (Theorem 3.3) to conclude that with probability at least $1-\exp(-4m)$,

$$\left|\frac{\sum_{e\in E, 1\leq \mathsf{i}\leq d} X_{e,\mathsf{i}}}{dm} - \mathsf{DICUT}(G,\rho_{L\sqcup R})\right| \leq \epsilon'/4\,.$$

Conditioned on $|\tilde{E}| \geq md(1-\epsilon'/8)$, we have

$$\left|\mathsf{DICUT}(\tilde{G}, L\sqcup R) - \mathsf{DICUT}(G,\rho_{L\sqcup R})\right| \leq \epsilon'/2\,.$$

Since $\mathsf{Max\text{-}DICUT}(G) = \max_{\rho:V\to\{0,1\}} \mathsf{DICUT}(G,\rho) = \sup_{\rho:V\to[0,1]} \mathsf{DICUT}(G,\rho)$, taking the union bound over the 2^{2m} possible partitions of $\tilde{V}$,[7] we conclude that with probability at least $1-\exp(-2m)$,

$$|\mathsf{Max\text{-}DICUT}(\tilde{G}) - \mathsf{Max\text{-}DICUT}(G)| \le \epsilon'/2\,.$$

The total number of edges incident on high-degree vertices in $\tilde{G}$. For a vertex $\tilde{v} \in \tilde{V}$, let $Y_{\tilde{v}} = \mathbb{1}[\deg(\tilde{v}) > 11d]\deg(\tilde{v})$. Then $Y = \sum_{\tilde{v}\in\tilde{V}} Y_{\tilde{v}}$ denotes the total sum of the degrees of vertices whose degree is larger than $11d$. It suffices to show that for any $\tilde{v}$, $\mathbb{E}[Y_{\tilde{v}}] < 1$. This is because applying Markov's inequality, we would get that $Y < 8m/7 < \epsilon'|\tilde{E}|/8$ with probability at least $7/8$, conditioned on $|\tilde{E}| \ge md(1-\epsilon'/8)$. Since $\mathsf{Max\text{-}DICUT}(\tilde{G}) \ge |\tilde{E}|/4$, the graph $\overline{G}$ obtained by deleting all the edges incident on vertices of degree larger than $11d$ in $\tilde{G}$ would satisfy $|\mathsf{Max\text{-}DICUT}(\overline{G}) - \mathsf{Max\text{-}DICUT}(\tilde{G})| \le \epsilon'/2$. We also get that $|\overline{E}| \ge md(1-\epsilon')/4$.

It remains to show that $\mathbb{E}[Y_{\tilde{v}}] < 1$. We will argue that for $d' > 11d$, $\Pr[\deg(\tilde{v}) \ge d'] \le 2^{-d'}$ (thus, $\mathbb{E}[Y_{\tilde{v}}] \le \sum_{d'>11d} \frac{d'}{2^{-d'}} < 1$). Consider any vertex $\tilde{v} = (v,i) \in V$. Consider any edge $e \in E$ that is incident on v. For $1 \le j \le d$, let $Z_{e,j}$ be the indicator random variable for the event that in the j-th iteration corresponding to e, the condition in Algorithm 1 is satisfied and the edge that is sampled is incident on $\tilde{v}$. We have

$$\deg(\tilde{v}) = \sum_{e\in E: e \text{ is incident on } v} \sum_{1\le j\le d} Z_{e,j}\,.$$

Observe that $\mathbb{E}[Z_{e,j}] \le \frac{1}{\deg(v)(1-\epsilon'/100)}$ Thus, by linearity of expectation, $\mathbb{E}[\deg(\tilde{v})] \le \frac{d\deg(v)}{\deg(v)(1-\epsilon'/100)} \le 2d$ and since the edges were sampled independently, we can apply Chernoff bound (Theorem 3.2) to conclude that $\Pr[\deg(\tilde{v}) \ge d'] \le 2^{-d'}$. ◀

5 Three-pass streaming algorithm for graphs of bounded average degree

In this section, for every constant $\epsilon > 0$, we give a three-pass streaming algorithm that achieves $(1/2-\epsilon)$-approximation on any n-vertex graph with average degree *at most* n^{δ}, using space $O(n^{1-\delta})$, for some $\delta \in (0,1)$.

Algorithm 2 Three-pass streaming algorithm for Max-DICUT on graphs of bounded average degree.

Parameters: $n \in \mathbb{N}$, $\epsilon \in (0, 1/2)$
Input: A stream $\boldsymbol{\sigma}$ of edges of directed graph $G(V,E)$ on n vertices and at most $n^{1+\delta}$ edges, where $\delta = \epsilon^{25/\epsilon}$

First pass:

Maintain a counter for the number of edges m
Set a parameter $\beta \triangleq \epsilon^{20/\epsilon}$
Initialize $V', E' = \emptyset$
Initialize an n-length array count (indexed by V) to be 0^n
for every edge $u \to v$ in the stream **do**

[7] In particular, consider the Max-DICUT partition $\rho^* : V \to \{0,1\}$ of G and the partition $L \sqcup R$ of $\tilde{V}$ where $(v,i) \in L$ iff $\rho^*(v) = 1$. By definition, $\rho_{L\sqcup R} = \rho^*$.

Sample $z_1, z_2 \sim \mathsf{Bern}(n^{-\beta})$ independently[a]

[a] We denote by $\mathsf{Bern}(p)$ the Bernoulli distribution where we draw 1 with probability p and 0 with probability $1-p$.

if $z_1 = 1$ **then**
  Add $(u, \mathsf{count}[u])$ to V' and increment $\mathsf{count}[u]$ by 1
if $z_2 = 1$ **then**
  Add $(v, \mathsf{count}[v])$ to V' and increment $\mathsf{count}[v]$ by 1
if $|V'| > n^{1-3\beta/4}$ **then**
  Terminate the execution

Second pass:

if $m \le n^{1-\epsilon^{1/\epsilon}}$ **then**
  Store all the edges and compute the Max-DICUT value.
else
  for every $v \in V$ such that $\mathsf{count}[v] > 0$ **do**
    Compute $\deg(v)$ in the stream
    $\forall i \in [\mathsf{count}[v]]$, initialize $d(v,i) = 0$

Third pass:

Set a parameter $d = 320/\epsilon^2$
for every edge $u \to v$ in the stream **do**
  Initialize $\mathsf{i} = 1$
  for $\mathsf{i} \le d$ **do**
    if $\mathsf{count}[u] > 0$ **then**
      Sample $i_1 \sim \mathsf{Unif}([\deg(u)])$
      if $i_1 < \mathsf{count}[u]$ **then**
        Increment $d(u, i_1)$ by 1
    if $\mathsf{count}[v] > 0$ **then**
      Sample $i_2 \sim \mathsf{Unif}([\deg(v)])$
      if $i_2 < \mathsf{count}[v]$ **then**
        Increment $d(v, i_2)$ by 1
    if $\mathsf{count}[u] > 0, \mathsf{count}[v] > 0$ and $i_1 < \mathsf{count}[u], i_2 < \mathsf{count}[v]$ **then**
      Add $(u, i_1) \to (v, i_2)$ to E'
      if $|E'| > n^{1-\epsilon^{25/\epsilon}}$ **then**
        Terminate the execution
    Increment i by 1

Post processing steps of Algorithm 2

1. For every $(u,i) \in V'$ such that $d(u,i) > 11d = 3520/\epsilon^2$, delete every edge incident on (u,i) in E' and decrement $d(\cdot)$ each of its endpoints by 1. Finally, set $d(u,i) = 0$. Let $\mathcal{G}(V', E')$ be the directed graph on vertex set V' and edge set E'.
2. Let $\ell, c \in \mathbb{N}$ be set according to Corollary 3.7 for parameter $\epsilon'' = \epsilon/8$. Let $D = 11d$. Let $\mathcal{H}_{2m}$ be the hash family in Corollary 3.7 corresponding to directed graphs on $2m$ vertices. Sample $h \sim \mathsf{Unif}(\mathcal{H}_{2m})$.
3. For $T \in \mathbf{Typ}_c^{\ell,D}$, let a_T denote the number of edges $e \in E'$ such that every vertex $(v,i) \in V'$ which is at a distance of at most $\ell - 1$ from at least one of the endpoints of e satisfies the property that $\deg_{\mathcal{G}}(v,i) = d(v,i)$, and $\mathsf{nbhdtype}^{\ell}_{\mathcal{G},h|_{V'}}(e) = T$.

4. Consider the distribution $\mathcal{D} \in \Delta(\mathbf{Typ}_c^{\ell,D})$, defined to be

$$\mathcal{D}(T) = \frac{a_T \cdot n^{\beta|T|}}{\sum_{T' \in \mathbf{Typ}_c^{\ell,D}} a_{T'} \cdot n^{\beta|T'|}},$$

where $T \in \mathbf{Typ}_c^{\ell,D}$ and $|T|$ denotes the number of vertices in T.
5. Let $\texttt{Local} : \mathbf{Typ}_c^{\ell,D} \to [0,1]$ be the function in Corollary 3.7. Output $\mathbb{E}_{T\sim\mathcal{D}}[\texttt{Local}(T)]$.

▶ **Theorem 5.1.** *For every $\epsilon \in (0,1/2)$, there exists $\mathsf{n}_0 \in \mathbb{N}$ such that for all $n \geq \mathsf{n}_0$ and every directed graph G on n vertices and $m \leq n^{1+\epsilon^{25/\epsilon}}$ edges, Algorithm 2 outputs a value v satisfying*

$$\frac{\mathsf{Max\text{-}DICUT}(G)}{2} - \epsilon \leq v \leq \mathsf{Max\text{-}DICUT}(G) + \epsilon,$$

with probability at least $2/3$, using space $n^{1-\epsilon^{O(1/\epsilon)}}$.

Proof. We first analyze the space usage of Algorithm 2 and then argue its correctness.

Space usage of Algorithm 2. It follows from Algorithm 2 that $|V'|, |E'| \leq n^{1-\epsilon^{O(1/\epsilon)}}$. In Algorithm 2, we can maintain count as an associative array with keys in V, thus storing only the non-zero entries in count, which is at most $|V'|$. We initialized count as an n-length array to keep the presentation simple.

Correctness of Algorithm 2. Consider the bounded-degree graph $\overline{G}$ that we constructed in Theorem 4.1 such that

$$\mathsf{Max\text{-}DICUT}(G) - \epsilon/2 \leq \mathsf{Max\text{-}DICUT}(\overline{G}) \leq \mathsf{Max\text{-}DICUT}(G) + \epsilon/2.$$

We will show that with probability at least $2/3$,

$$\mathsf{tvd}(\mathcal{D}, \mathsf{EdgeNbhdTypeDist}^{\ell}_{\overline{G};h}) \leq \epsilon/8.$$

Therefore, combining Theorem 4.1 and Corollaries 3.7 and 3.8, the correctness of Algorithm 2 immediately follows. Let us first recall the construction of $\overline{G}$ in Algorithm 1 for the parameter $\epsilon' = \epsilon/2$, and when $\mathsf{est\text{-}deg}[v] = \deg(v)$ for every $v \in V$. For every vertex $v \in V$, we add $\deg(v)$ many copies of it to $\overline{V}$, i.e., $\overline{V} = \{(v,i) : v \in V, i \in [\deg(v)]\}$. We fix a parameter $d = 10/(\epsilon')^2 = 40/\epsilon^2$. For every edge $u \to v \in E$, we sample a copy (u,i) of u and a copy (v,j) of v uniformly at random from $\overline{V}$, and add the edge $(u,i) \to (v,j)$ to $\overline{E}$. We repeat this process d times independently for every edge, and finally, for every vertex of degree larger than $11d$ in $\overline{V}$, we delete all the edges incident to it. We prove that with probability at least $5/6$,

$$\mathsf{Max\text{-}DICUT}(G) - \epsilon/2 \leq \mathsf{Max\text{-}DICUT}(\overline{G}) \leq \mathsf{Max\text{-}DICUT}(G) + \epsilon/2.$$

For now, let us ignore the termination conditions in Algorithm 2. We will later argue that these events occur with very low probability and hence, do not affect the success probability of the algorithm by much. We claim the following.

▷ Claim 5.2. At the end of the third pass (before post processing), the following hold:
1. V' has the same distribution as sampling every vertex from $\overline{V}$ independently with probability $n^{-\beta}$, and applying a relabeling on the vertices, and

2. for $u \to v \in E$, if both u and v have copies in V', then the copies of $u \to v$ in $\tilde{G}$ are sampled identically in Algorithms 1 and 2. If only one of u or v has a copy in V', then the endpoints corresponding to the copies of that vertex in the copies of $u \to v$ in $\tilde{G}$ are sampled identically in Algorithms 1 and 2.

Proof. Consider the map $f : \overline{V} \to E$ defined as follows: for $(v, i) \in \overline{V}$, $f(v, i) = e$, where e is the $(i + 1)$-th edge incident on v in the stream $\boldsymbol{\sigma}$. Observe that for every edge $u \to v$, there are exactly two vertices in $\overline{V}$ that map to it, namely (u, i) and (v, j) where i, j are such that e is the $(i + 1)$-th edge (resp. $(j + 1)$-th edge) incident on u (resp. v) in the stream $\boldsymbol{\sigma}$. Therefore, sampling every vertex in $\overline{V}$ independently with probability $n^{-\beta}$ is the same as sampling each vertex in $f^{-1}(e)$, for every edge e, independently with probability $n^{-\beta}$. This is exactly what we do in Algorithm 2 in Algorithm 2. However, since we cannot maintain information about the streaming order of all the edges incident on every vertex in V, we relabel the copies of v in $\overline{V}$ based on the order in which they are sampled by our algorithm. In particular, for a vertex $v \in V$, the first copy that is sampled is labeled $(v, 0)$, the second copy that is sampled is labeled $(v, 1)$, and so on. This proves Item 1 of Claim 5.2. To see Item 2 of Claim 5.2, recall how the edges were sampled for $\tilde{G}$ in Algorithm 1. For our purposes, it suffices to consider the sampling of the copies of only those edges in E that are incident on a vertex $v \in V$ which has a copy in V', i.e., $V' \cap \{(v, i) : i \in [\deg(v)]\} \neq \emptyset$. Algorithm 2 check for this condition and if they are satisfied, the sampling is identical to Algorithm 1 of Algorithm 1. For an edge $u \to v$, its copy is added to E' in Algorithm 2 iff the sampled copies of both its endpoints are in V'. Observe that for $(v, i) \in V'$, all the edges incident on v in E satisfy either the condition in Algorithm 2 or Algorithm 2. ◁

Given the above claim, we consider $\tilde{G}$ where for $u \to v \in E$, if both u and v have copies in V', then the copies of $u \to v$ are sampled according to Algorithm 2 (prior to the post-processing steps). If only one of u or v has a copy in V', then the endpoints corresponding to the copies of that vertex in the copies of $u \to v$ are sampled according to Algorithm 2 and the other endpoints are sampled independently, according to Algorithm 1. The remaining edges are also sampled according to Algorithm 1. Finally, like in Algorithm 1, we define $\overline{G}$ as $\tilde{G}$ with edges incident on vertices of degree larger than $11d$ deleted.[8] Observe that for $\overline{G}$ as defined above, $\mathcal{G}(V', E')$ is the induced subgraph of $\overline{G}$ on V' and furthermore, at the end of the post-processing steps of Algorithm 2, for $(v, i) \in V'$, $d(v, i)$ is degree of (v, i) in $\overline{G}$. This is true because for any vertex $(v, i) \in V'$, its degree in $\tilde{G}$ depends only on the randomness in Algorithm 2.

We are now ready to show that with probability at least 3/4,

$$\mathsf{Max\text{-}DICUT}(G) - \epsilon/2 \leq \mathsf{Max\text{-}DICUT}(\overline{G}) \leq \mathsf{Max\text{-}DICUT}(G) + \epsilon/2\,,$$

and

$$\mathsf{tvd}(\mathcal{D}, \mathsf{EdgeNbhdTypeDist}^{\ell}_{\overline{G};h}) \leq \epsilon/8\,.$$

We apply Theorems 3.9 and 4.1. In particular, for Theorem 4.1, we set the parameter $\epsilon' = \epsilon/2$. We get that with probability at least 5/6,

- $\mathsf{Max\text{-}DICUT}(G) - \epsilon/2 \leq \mathsf{Max\text{-}DICUT}(\overline{G}) \leq \mathsf{Max\text{-}DICUT}(G) + \epsilon/2$,
- the maximum degree of any vertex in $\overline{G}$ is at most $110/(\epsilon')^2 = 3520/\epsilon^2$, and
- $\overline{G}$ has at least $240m/\epsilon^2$ edges.

[8] A key observation here is that sampling $\overline{G}$ first and then sampling V' is equivalent to sampling V' and then sampling $\overline{G}$.

Since $\overline{G}$ has exactly $2m$ vertices, it follows that at most 0.94 fraction of them are isolated. Now conditioned on this $\overline{G}$, we set the parameters of Theorem 3.9. We set $\epsilon'' = \epsilon/8, D = 3520/\epsilon^2$, and c, ℓ, h according to Algorithm 2, and $\alpha = 0.94$. Now set β_0 and n_0 according to the proof of Theorem 3.9. It follows from Claim 5.2 that V' is obtained by sampling every vertex in $\overline{V}$ independently with probability $n^{-\beta}$, and $\mathcal{G}(V', E')$ is the induced subgraph of $\overline{G}$ on V'. Say $2m \geq n_0$. To apply Theorem 3.9, observe that sampling $\overline{G}$ first and then sampling V' is the same as sampling V' first and then sampling $\overline{G}$, and relabeling of the copies of the vertices does not the change the distribution. Thus, it remains to show the following.

1. The sampling probability $n^{-\beta}$ is at least $(2m)^{-\beta_0}$, and
2. a_T defined in the third post-processing step of Algorithm 2 is the same as the a_T defined in Theorem 3.9.

If $m \leq n^{1-\epsilon^{1/\epsilon}}$, then Algorithm 2 would simply store all the edges and compute the Max-DICUT value. So without loss of generality, we can assume that $m > n^{1-\epsilon^{1/\epsilon}}$, and in this case, $n^{-\beta} \geq (2m)^{-\beta_0}$. In Claim 5.2, we also show that for every $(v, i) \in V'$, $d(v, i)$ is the degree of (v, i) in $\overline{G}$. Hence, for any edge $e \in E'$, its radius-ℓ neighborhood in $\overline{G}$ is contained entirely within $\mathcal{G}$ iff for every vertex $(v, i) \in V'$ which is at a distance of at most $\ell - 1$ from at least one of the endpoints of e satisfies $\deg_{\mathcal{G}}(v, i) = d(v, i)$. It follows that the a_T defined in the post-processing steps of Algorithm 2 is the same as the one in Theorem 3.9. Thus, conditioned on $\overline{G}$ satisfying the properties that we stated above and $2m \geq n_0$, with probability at least 9/10,

$$\mathsf{tvd}(\mathcal{D}, \mathsf{EdgeNbhdTypeDist}^{\ell}_{\overline{G};h}) \leq \epsilon/8\,.$$

Set $\mathsf{n}_0 = \max\left\{n_0^{(1-\epsilon^{1/\epsilon})^{-1}}, (\frac{1}{\epsilon^2})^{\frac{1}{\epsilon}\frac{25}{\epsilon}}\right\}$. Thus, whenever $n \geq \mathsf{n}_0$, we have $m \geq n_0$.

We finally argue that the termination conditions in Algorithm 2 happen with very low probability. Since $m \leq n^{1+\epsilon^{25/\epsilon}}$ and $|\overline{V}| = 2m$, we have $\mathbb{E}[V'] \leq n^{1-7\beta/8}$. By applying Chernoff bound, we conclude that with very high probability, $|V'| \leq n^{1-3\beta/4}$. For an edge $e \in \overline{E}$, let Z_e be the indicator random variable for the event that $e \in E'$. Let $Z = \sum_{e \in \overline{E}} Z_e$. We have $\mathbb{E}[Z_e] = n^{-2\beta} = n^{-2\epsilon^{20/\epsilon}}$ and $\mathsf{Var}[Z] \leq (3520/\epsilon^2)\mathbb{E}[Z]$, since the maximum degree of any vertex in $\overline{G}$ is at most $3520/\epsilon^2$. Since $|\overline{E}| \leq 320m/\epsilon^2$, $\mathbb{E}[Z] \leq n^{1-\epsilon^{22/\epsilon}}$. By applying Chebyshev's inequality, we conclude that with high probability, $|E'| \leq n^{1-\epsilon^{25/\epsilon}}$. This concludes the proof of correctness of Algorithm 2. ◀

6 Two-pass streaming algorithm for Max-DICUT

In this section, we modify[9] Algorithm 2 to give a two-pass streaming algorithm that achieves $(1/2 - \epsilon)$-approximation on any n-vertex graph with average degree *at most* n^{δ}, using space $O(n^{1-\delta})$, for some $\delta \in (0, 1)$. Combining our algorithm with the two-pass algorithm of Bhaskara, Daruki, and Venkatasubramanian [5] (see Theorem 2.3) that achieves for every $\epsilon, \delta \in (0, 1)$, $(1 - \epsilon)$-approximation for graphs of average degree *at least* n^{δ} using space $\tilde{O}(n^{1-\delta})$, we obtain our algorithm that works on arbitrary graphs (see Algorithm 4).

Algorithm 3 Two-pass streaming algorithm for Max-DICUT on graphs of bounded average degree.

Parameters: $n \in \mathbb{N}$, $\epsilon \in (0, 1/2)$

Input: A stream $\boldsymbol{\sigma}$ of edges of directed graph $G(V, E)$ on n vertices and at most $n^{1+\delta}$ edges, where $\delta = \epsilon^{25/\epsilon}$

[9] The specific modifications are highlighted in the code.

First pass:

Maintain a counter for the number of edges m
Set a parameter $\beta \triangleq \epsilon^{20/\epsilon}$
Initialize $V', E' = \emptyset$
Initialize an n-length array count (indexed by V) to be 0^n
Initialize an n-length array est-deg (indexed by V) to be 0^n
for every edge $u \to v$ in the stream **do**
    Sample $z_1, z_2 \sim \mathsf{Bern}(n^{-\beta})$ independently
    if $z_1 = 1$ **then**
        Add $(u, \mathsf{count}[u])$ to V' and increment $\mathsf{count}[u]$ by 1
    if $z_2 = 1$ **then**
        Add $(v, \mathsf{count}[v])$ to V' and increment $\mathsf{count}[v]$ by 1
    if $|V'| > n^{1-3\beta/4}$ **then**
        Terminate the execution
    Sample $y_1, y_2 \sim \mathsf{Bern}(n^{-\beta/4})$ independently
    if $y_1 = 1$ **then**
        Increment $\mathsf{est\text{-}deg}[u]$ by $n^{\beta/4}$
    if $y_2 = 1$ **then**
        Increment $\mathsf{est\text{-}deg}[v]$ by $n^{\beta/4}$
    if # non-zero entries in est-deg $> n^{1-\beta/8}$ **then**
        Terminate the execution

for every $v \in V$ such that $\mathsf{count}[v] > 0$ **do**
    $\forall i \in [\mathsf{count}[v]]$, initialize $d(v, i) = 0$
Initialize an n-length array store-deg (indexed by v) to be 0^n
Initialize $\hat{E} = \emptyset$

Second pass:

if $m \le n^{1-\epsilon^{1/\epsilon}}$ **then**
    Store all the edges and compute the Max-DICUT value.
else
    Set a parameter $d = 320/\epsilon^2$
    for every edge $u \to v$ in the stream **do**
        if $\mathsf{count}[u] > 0$ and $\mathsf{store\text{-}deg}[u] < n^{2\beta/3}$ **then**
            Add $u \to v$ to $\hat{E}$
            Increment $\mathsf{store\text{-}deg}[u]$
        if $\mathsf{count}[v] > 0$ and $\mathsf{store\text{-}deg}[v] < n^{2\beta/3}$ **then**
            Add edge $u \to v$ to $\hat{E}$ if it was not previously added in Algorithm 3
            Increment $\mathsf{store\text{-}deg}[v]$
        Initialize $\mathsf{i} = 1$
        for $\mathsf{i} \le d$ **do**
            if $\mathsf{count}[u] > 0$ and $\mathsf{est\text{-}deg}(u) > 0$ **then**
                Sample $i_1 \sim \mathsf{Unif}([\mathsf{est\text{-}deg}(u)])$
                if $i_1 < \mathsf{count}[u]$ **then**
                    Increment $d(u, i_1)$ by 1
            if $\mathsf{count}[v] > 0$ and $\mathsf{est\text{-}deg}(v) > 0$ **then**

Sample $i_2 \sim \mathsf{Unif}([\mathsf{est\text{-}deg}(v)])$
if $i_2 < \mathsf{count}[v]$ **then**
Increment $d(v, i_2)$ by 1
if $(\mathsf{count}[u] > 0, \mathsf{est\text{-}deg}(u) > 0, i_1 < \mathsf{count}[u])$ or $(\mathsf{count}[v] > 0, \mathsf{est\text{-}deg}(v) > 0, i_2 < \mathsf{count}[v])$ **then**
Add $(u, i_1) \to (v, i_2)$ to E'
if $|E'| > n^{1-\epsilon^{25/\epsilon}}$ **then**
Terminate the execution
Increment i by 1

Post-processing I: Resampling edges incident to low-degree vertices:
Initialize $E'' = E'$
for every $u \in V$ such that $\mathsf{count}[u] > 0$ and $\mathsf{store\text{-}deg}[u] < n^{2\beta/3}$ **do**
Initialize $i = 0$
for $i \in [\mathsf{est\text{-}deg}[u]]$ **do**
for every edge $(u, i) \to (v, j) \in E''$ or $(v, j) \to (u, i) \in E''$ **do**
Delete the edge; decrement $d(u, i)$ (resp. $d(v, j)$) if $(u, i) \in V'$ (resp. $(v, j) \in V'$)
for every $u \to v \in \hat{E}$ **do**
Initialize j = 1
for j $\leq d$ **do**
if $\mathsf{count}[u] > 0$ and $\mathsf{store\text{-}deg}(u) < n^{2\beta/3}$ **then**
Sample $j_1 \sim \mathsf{Unif}([\mathsf{store\text{-}deg}(u)])$
if $j_1 < \mathsf{count}[u]$ **then**
Increment $d(u, j_1)$ by 1
if $\mathsf{count}[v] > 0$ **then**
if $\mathsf{store\text{-}deg}(v) < n^{2\beta/3}$ **then**
Sample $j_2 \sim \mathsf{Unif}([\mathsf{store\text{-}deg}(v)])$
else
Sample $j_2 \sim \mathsf{Unif}([\mathsf{est\text{-}deg}(v)])$
if $j_2 < \mathsf{count}[v]$ **then**
Increment $d(v, j_2)$ by 1
if $j_1 < \mathsf{count}[u]$ **then**
Add $(u, j_1) \to (v, j_2)$ to E''
else if $\mathsf{count}[v] > 0$ and $\mathsf{store\text{-}deg}(v) < n^{2\beta/3}$ **then**
Sample $j_2 \sim \mathsf{Unif}([\mathsf{store\text{-}deg}(v)])$
if $j_2 < \mathsf{count}[v]$ **then**
Increment $d(v, j_2)$ by 1
if $\mathsf{count}[u] > 0$ **then**
Sample $j_1 \sim \mathsf{Unif}([\mathsf{est\text{-}deg}(u)])$
if $j_1 < \mathsf{count}[u]$ **then**
Increment $d(u, j_1)$ by 1
if $j_2 < \mathsf{count}[v]$ **then**
Add $(u, j_1) \to (v, j_2)$ to E''
Increment j by 1
Delete every edge in E'' that is incident on a vertex not in V'

Post-processing II (analogous to the post processing steps in Algorithm 2)

1. For every $(u,i) \in V'$ such that $d(u,i) > 11d = 3520/\epsilon^2$, delete every edge incident on (u,i) in E'' and decrement $d(\cdot)$ each of its endpoints by 1. Finally, set $d(u,i) = 0$. Let $\mathcal{G}(V', E'')$ be the directed graph on vertex set V' and edge set E''.
2. Let $\ell, c \in \mathbb{N}$ be set according to Corollary 3.7 for parameter $\epsilon'' = \epsilon/8$. Let $D = 11d$. Let $\mathcal{H}_{2m}$ be the hash family in Corollary 3.7 corresponding to directed graphs on $2m$ vertices. Sample $h \sim \mathsf{Unif}(\mathcal{H}_{2m})$.
3. For $T \in \mathbf{Typ}_c^{\ell,D}$, let a_T denote the number of edges $e \in E''$ such that every vertex $(v,i) \in V'$ which is at a distance of at most $\ell - 1$ from at least one of the endpoints of e satisfies the property that $\deg_{\mathcal{G}}(v,i) = d(v,i)$, and $\mathsf{nbhdtype}^{\ell}_{\mathcal{G},h|_{V'}}(e) = T$.
4. Consider the distribution $\mathcal{D} \in \Delta(\mathbf{Typ}_c^{\ell,D})$, defined as

$$\mathcal{D}(T) = \frac{a_T \cdot n^{\beta|T|}}{\sum_{T' \in \mathbf{Typ}_c^{\ell,D}} a_{T'} \cdot n^{\beta|T'|}},$$

   where $T \in \mathbf{Typ}_c^{\ell,D}$, and $|T|$ denotes the number of vertices in T.
5. Let $\mathtt{Local} : \mathbf{Typ}_c^{\ell,D} \to [0,1]$ be the function in Corollary 3.7. Output $\mathbb{E}_{T\sim\mathcal{D}}[\mathtt{Local}(T)]$.

▶ **Theorem 6.1.** *For every $\epsilon \in (0,1/2)$, there exists $\mathsf{n}_0 \in \mathbb{N}$ such that for all $n \geq \mathsf{n}_0$ and every directed graph G on n vertices and $m \leq n^{1+\epsilon^{25/\epsilon}}$ edges, Algorithm 3 outputs a value v satisfying*

$$\frac{\mathsf{Max\text{-}DICUT}(G)}{2} - \epsilon \leq v \leq \mathsf{Max\text{-}DICUT}(G) + \epsilon,$$

with probability at least $2/3$, using space $n^{1-\epsilon^{O(1/\epsilon)}}$.

Proof. At a high level, the main change from Algorithm 2 is that for vertices randomly sampled from $\overline{V}$ in the first-pass, we don't use an additional pass like in Algorithm 2 to exactly compute the degrees of all their parent vertices. Instead, in the first pass, we approximately estimate the degrees of all high-degree vertices (of degree at least $n^{2\beta/3}$) with high probability. As shown in Theorem 4.1, for high-degree vertices, an approximate estimate of their degrees suffices for the reduction to work. In the second pass, we handle the low degree vertices and store all the edges incident to parent vertices of degree less than $n^{2\beta/3}$ and resample their copies in $\overline{G}$ in the post-processing steps from Algorithm 3 to Algorithm 3. In addition, for technical reasons, E' now stores *all* edges incident on vertices in V', not just the induced edges (see Algorithm 3). This helps us maintain accurate degrees of the sampled vertices when we delete the edges sampled in the second pass that were incident to copies of low degree vertices and resample them later in post-processing I. We now formally analyze the space usage and argue the correctness of Algorithm 3.

Space usage of Algorithm 3. Compared to Algorithm 2, Algorithm 3 also stores est-deg, store-deg, and $\hat{E}$. It follows from the termination condition in Algorithm 3 that est-deg has at most $n^{1-\beta/8}$ non-zero entries, and the value stored in each entry is at most $\mathrm{poly}(n)$. Thus, the space required to store est-deg is at most $n^{1-\epsilon^{O(1/\epsilon)}}$. The termination condition in Algorithm 3 ensures that $|V'| \leq n^{1-3\beta/4}$. The number of non-zero entries in store-deg is at most $|V'|$. Finally, whenever an edge is added to $\hat{E}$, both its endpoints have copies in V' and at least one of them currently has fewer than $n^{2\beta/3}$ edges incident on it in $\hat{E}$. Therefore, the number of edges stored in $\hat{E}$ is at most $|V'|n^{2\beta/3} \leq n^{1-\beta/12}$.

Correctness of Algorithm 3. For now, let us ignore the termination conditions in Algorithm 3. We will later argue that these events occur with very low probability, and hence, do not alter the success probability of Algorithm 3 by much. We first prove the following claim.

▷ Claim 6.2. Let n be sufficiently large. With probability at least $1 - \exp(n^{-\Omega(\epsilon^{O(1/\epsilon)})})$, for every $v \in V$ such that $\deg(v) \geq n^{2\beta/3}$, we have

$$(1 - \epsilon/100)\deg(v) \leq \textsf{est-deg}[v] \leq (1 + \epsilon/100)\deg(v)\,.$$

Proof. Fix any vertex $v \in V$ whose degree is at least $n^{2\beta/3}$. For $i \in [\deg(v)]$, let $Y_i \sim \textsf{Bern}(n^{-\beta/4})$ independently. Then, $\textsf{est-deg}[v] = \sum_{i\in[\deg(v)]} Y_i \cdot n^{\beta/4}$. Therefore, $\mathbb{E}[\textsf{est-deg}[v]] = \deg(v)$. By applying Hoeffding's inequality (Theorem 3.3), we conclude that with probability at least $1 - \exp(n^{-\Omega(\epsilon^{O(1/\epsilon)})})$,

$$(1 - \epsilon/100)\deg(v) \leq \textsf{est-deg}[v] \leq (1 + \epsilon/100)\deg(v)\,.$$

The claim follows from taking a union bound over at most n vertices. ◁

Now let us condition on est-deg that satisfies the above property. We define an n-length array approx-deg as follows. For every vertex $v \in V$, if $\deg(v) < n^{2\beta/3}$, then $\textsf{approx-deg}[v] := \deg(v)$, else $\textsf{approx-deg}[v] := \textsf{est-deg}[v]$. Consider the bounded-degree graph $\overline{G}(\overline{V}, \overline{E})$[10] that is output by Algorithm 1, corresponding to the input graph $G(V, E)$, parameters $\epsilon' = \epsilon/2$ and $\zeta = 2\beta/3$, and approx-deg. It follows from Theorem 4.1 that with probability at least 5/6,

- $\textsf{Max-DICUT}(G) - \epsilon/2 \leq \textsf{Max-DICUT}(\overline{G}) \leq \textsf{Max-DICUT}(G) + \epsilon/2$,
- the maximum degree of any vertex in $\overline{G}$ is at most $110/(\epsilon')^2 = 3520/\epsilon^2$, and
- $\overline{G}$ has exactly $2m$ vertices and at least $240m/\epsilon^2$ edges.

We will prove the following claim (analogous to Claim 5.2 in Theorem 5.1).

▷ Claim 6.3. At the end of post-processing I, the following hold:

1. V' has the same distribution as sampling every vertex from $\overline{V}$ independently with probability $n^{-\beta}$, and applying a relabeling on the vertices, and
2. for $u \to v \in E$, if both u and v have copies in V', then the copies of $u \to v$ in $\tilde{G}$ are sampled identically in Algorithms 1 and 3. If only one of u or v has a copy in V', then the endpoints corresponding to the copies of that vertex in the copies of $u \to v$ in $\tilde{G}$ are sampled identically in Algorithms 1 and 3.

Proof. The proof of Item 1 of Claim 6.3 is analogous to the proof of Item 1 of Claim 5.2. To prove Item 2, we first consider the edges incident to high-degree vertices which have copies in V'. Since approx-deg and est-deg are the same for vertices of degree at least $n^{2\beta/3}$, the sampling in Algorithm 3 of Algorithm 3 is the same as the sampling in Algorithm 1 of Algorithm 1 for edges whose both endpoints are of degree at least $n^{2\beta/3}$ in G. Similar, if only one of the endpoints of $u \to v$ has a copy in V' and if the parent vertex of that copy is of degree at least $n^{2\beta/3}$, then the the endpoints corresponding to the copies of that vertex in the copies of $u \to v$ in $\tilde{G}$ are sampled identically in Algorithms 1 and 3. Now consider the edges incident to low-degree vertices which have copies in V'. While these edges are **not** sampled with the correct probabilities in Algorithm 3, this is fixed in the post processing steps

[10] We implicitly assume that Algorithm 1 uses the same randomness as in Algorithm 3 to sample copies of the edges incident on the parent vertices of vertices in V'. To sample the remaining edges, it uses independent random bits.

from Algorithm 3 to Algorithm 3. For every $(u,i) \in V'$, observe that $\textsf{store-deg}[u] < n^{2\beta/3}$ if and only if $\deg(u) < n^{2\beta/3}$. In addition, when $\deg(u) < n^{2\beta/3}$, $\textsf{store-deg}[u] = \deg[u]$. In post-processing I, for every low-degree vertex that has a copy in V', we delete all edges in E'' that are incident on *any* of its copies, including those not contained in V'. All edges incident to low-degree vertices (which have a copy in V') are stored in $\hat{E}$. Since $\textsf{store-deg}[u] = \deg[u]$ for these vertices, we resample the copies of the edges incident on them identical to the sampling in Algorithm 1. ◁

Given the above claim, we consider $\tilde{G}$ where for $u \to v \in E$, if both u and v have copies in V', then the copies of $u \to v$ are sampled according to Algorithm 3 (prior to post-processing II). If only one of u or v has a copy in V', then the endpoints corresponding to the copies of that vertex in the copies of $u \to v$ are sampled according to Algorithm 3 and the other endpoints are sampled independently, according to Algorithm 1. The remaining edges are also sampled according to Algorithm 1. Finally, like in Algorithm 1, we define $\overline{G}$ as $\tilde{G}$ with edges incident on vertices of degree larger than $11d$ deleted. Observe that for $\overline{G}$ as defined above, $\mathcal{G}(V', E'')$ is the induced subgraph of $\overline{G}$ on V'. Furthermore, at the end of post-processing II of Algorithm 3, for $(v,i) \in V'$, $d(v,i)$ is degree of (v,i) in $\overline{G}$. Now, the rest of the proof of correctness follows similarly as in the proof of Theorem 5.1.

Finally, we argue that the termination conditions in Algorithm 3 happen with very low probability. Observe that V' is sampled exactly in the same way as in Algorithm 2. Therefore, the same proof goes through. To analyze the number of non-zero entries in $\textsf{est-deg}$, for $i \in [\log n]$, consider any vertex $v \in V$ whose degree is larger than 2^i and at most 2^{i+1}. Let Z^i_v be the indicator variable for the event that $\textsf{est-deg}[v] \neq 0$ and let $Z^i = \sum_{v:2^i < \deg(v) \leq 2^{i+1}} Z^i_v$. Observe that $\mathbb{E}[Z^i_v] \leq 2^{i+1} n^{-\beta/4}$ and by the linearity of expectation,

$$\mathbb{E}[Z^i] \leq \frac{2m}{2^i} \cdot 2^{i+1} n^{-\beta/4} = 4mn^{-\beta/4} \leq n^{1-\beta/6}\,.$$

Since the events are independent, applying Chernoff bound (Theorem 3.2), we conclude that with probability at least $1 - \exp(n^{-\Omega(\epsilon^{O(1/\epsilon)})})$, $Z^i \leq (\log n) n^{1-\beta/6}$. Taking a union bound over all $i \in [\log n]$, we get that with high probability, $\sum_{i \in [\log n]} Z^i \leq (\log n)^2 n^{1-\beta/6} < n^{1-\beta/8}$. We now analyze the termination condition on E' in Algorithm 3. Let E'_ℓ denote the set of edges in E' which are incident to vertices in V' whose parent vertices have degree less than $n^{2\beta/3}$. Let E'_h denote the set of edges in E' which are incident to vertices in V' whose parent vertices have degree at least $n^{2\beta/3}$. Observe that $E' = E'_\ell \cup E'_h$. We previously argued (in Section 5) that with high probability, $|V'| \leq n^{1-3\beta/4}$. Therefore, with high probability, $|E'_\ell| \leq n^{1-\beta/12}$. It follows from Claim 6.2 that with high probability, for every $v \in V$ such that $\deg(v) \geq n^{2\beta/3}$, we have

$$(1 - \epsilon/100)\deg(v) \leq \textsf{est-deg}[v] \leq (1 + \epsilon/100)\deg(v)\,.$$

Therefore, conditioned on this event, $\mathbb{E}[|E'_h|] \leq dn^{1+\delta}(2n^{-\beta}(1-\epsilon/100)^{-1}) \leq n^{1-3\beta/4}$, for sufficiently large n. Applying Markov's inequality, we conclude that with high probability, $|E'_h| \leq n^{1-2\beta/3}$. Thus, the termination condition on E' in Algorithm 3 occurs with very low probability, and this concludes the proof of correctness of Algorithm 3. ◀

Below, we describe our algorithm for Max-DICUT on arbitrary graphs.

Algorithm 4 Two-pass streaming algorithm for Max-DICUT.

Parameters: $n \in \mathbb{N}$, $\epsilon > 0$
Input: A stream $\boldsymbol{\sigma}$ of edges of a directed graph $G(V, E)$ on n vertices

First pass:
Maintain a counter for the number of edges m and execute the first passes of Algorithm 3 and the algorithm in [5], in parallel

Second pass:
if $m \leq n^{1+\epsilon^{4/\epsilon}}$ **then**
    Continue executing the second pass of Algorithm 3 and abandon the execution of [5]
else
    Abandon the execution of Algorithm 3 and execute the second pass of [5]

Finally, we conclude with a proof of our main theorem.

Proof of Theorem 1.1. The proof of correctness and the space usage of Algorithm 4 immediately follows from Theorem 6.1 and Theorem 2.3. ◀

References

1 Kook Jin Ahn and Sudipto Guha. Graph sparsification in the semi-streaming model. In *Proceedings of the 36th Internatilonal Collogquium on Automata, Languages and Programming: Part II*, ICALP '09, pages 328–338, Berlin, Heidelberg, 2009. Springer-Verlag. doi:10.1007/978-3-642-02930-1_27.

2 Noga Alon, Wenceslas Fernandez de la Vega, Ravi Kannan, and Marek Karpinski. Random sampling and approximation of max-csps. *J. Comput. Syst. Sci.*, 67(2):212–243, 2003. doi:10.1016/S0022-0000(03)00008-4.

3 Sepehr Assadi and Vishvajeet N. Graph streaming lower bounds for parameter estimation and property testing via a streaming XOR lemma. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 612–625. ACM, 2021. doi:10.1145/3406325.3451110.

4 Boaz Barak, Moritz Hardt, Thomas Holenstein, and David Steurer. Subsampling mathematical relaxations and average-case complexity. In Dana Randall, editor, *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 512–531. SIAM, 2011. doi:10.1137/1.9781611973082.41.

5 Aditya Bhaskara, Samira Daruki, and Suresh Venkatasubramanian. Sublinear algorithms for MAXCUT and correlation clustering. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 16:1–16:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.16.

6 Keren Censor-Hillel, Rina Levy, and Hadas Shachnai. Fast distributed approximation for max-cut. In Antonio Fernández Anta, Tomasz Jurdzinski, Miguel A. Mosteiro, and Yanyong Zhang, editors, *Algorithms for Sensor Systems - 13th International Symposium on Algorithms and Experiments for Wireless Sensor Networks, ALGOSENSORS 2017, Vienna, Austria, September 7-8, 2017, Revised Selected Papers*, volume 10718 of *Lecture Notes in Computer Science*, pages 41–56. Springer, 2017. doi:10.1007/978-3-319-72751-6_4.

7 Chi-Ning Chou, Alexander Golovnev, Madhu Sudan, Ameya Velingker, and Santhoshini Velusamy. Linear space streaming lower bounds for approximating csps. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 275–288. ACM, 2022. doi:10.1145/3519935.3519983.

8 Chi-Ning Chou, Alexander Golovnev, Madhu Sudan, and Santhoshini Velusamy. Approximability of all boolean csps with linear sketches, 2022. arXiv:2102.12351.

9 Chi-Ning Chou, Alexander Golovnev, Madhu Sudan, and Santhoshini Velusamy. Sketching approximability of all finite csps. *J. ACM*, 71(2):15:1–15:74, 2024. doi:10.1145/3649435.

10 Chi-Ning Chou, Alexander Golovnev, and Santhoshini Velusamy. Optimal streaming approximations for all boolean max-2csps and max-ksat. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 330–341. IEEE, 2020. doi:10.1109/FOCS46700.2020.00039.

11 Yumou Fei, Dor Minzer, and Shuo Wang. Multi-pass streaming lower bounds for approximating max-cut. In *66th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2025, Sydney, Australia, December 14-17, 2025*, pages 1537–1560. IEEE, 2025. doi:10.1109/FOCS63196.2025.00081.

12 Yumou Fei, Dor Minzer, and Shuo Wang. A dichotomy theorem for multi-pass streaming csps. In Aditya Bhaskara and Artur Czumaj, editors, *Proceedings of the 58th Annual ACM Symposium on Theory of Computing, STOC 2026, Salt Lake City, UT, USA, June 22-26, 2026*, pages 246–255. ACM, 2026. doi:10.1145/3798129.3800744.

13 Uriel Feige and Gideon Schechtman. On the optimality of the random hyperplane rounding technique for MAX CUT. *Random Struct. Algorithms*, 20(3):403–440, 2002. doi:10.1002/RSA.10036.

14 Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995. doi:10.1145/227683.227684.

15 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998. doi:10.1145/285055.285060.

16 Venkatesan Guruswami, Ameya Velingker, and Santhoshini Velusamy. Streaming complexity of approximating max 2csp and max acyclic subgraph. In Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, volume 81 of *LIPIcs*, pages 8:1–8:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.8.

17 Michael Kapralov, Sanjeev Khanna, and Madhu Sudan. Streaming lower bounds for approximating MAX-CUT. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1263–1282. SIAM, 2015. doi:10.1137/1.9781611973730.84.

18 Michael Kapralov, Sanjeev Khanna, Madhu Sudan, and Ameya Velingker. $(1 + \Omega(1))$-approximation to MAX-CUT requires linear space. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1703–1722. SIAM, 2017. doi:10.1137/1.9781611974782.112.

19 Michael Kapralov and Dmitry Krachun. An optimal space lower bound for approximating MAX-CUT. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 277–288. ACM, 2019. doi:10.1145/3313276.3316364.

20 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for MAX-CUT and other 2-variable csps? *SIAM J. Comput.*, 37(1):319–357, 2007. doi:10.1137/S0097539705447372.

21 Claire Mathieu and Warren Schudy. Yet another algorithm for dense max cut: go greedy. In Shang-Hua Teng, editor, *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2008, San Francisco, California, USA, January 20-22, 2008*, pages 176–182. SIAM, 2008. URL: http://dl.acm.org/citation.cfm?id=1347082.1347102.

22 Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 245–254. ACM, 2008. doi:10.1145/1374376.1374414.

23 Raghuvansh R. Saxena, Noah Singer, Madhu Sudan, and Santhoshini Velusamy. Streaming complexity of csps with randomly ordered constraints. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 4083–4103. SIAM, 2023. doi:10.1137/1.9781611977554.CH156.

24 Raghuvansh R. Saxena, Noah G. Singer, Madhu Sudan, and Santhoshini Velusamy. Improved streaming algorithms for maximum directed cut via smoothed snapshots. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 855–870. IEEE, 2023. doi:10.1109/FOCS57990.2023.00055.

25 Raghuvansh R. Saxena, Noah G. Singer, Madhu Sudan, and Santhoshini Velusamy. Streaming algorithms via local algorithms for maximum directed cut. In Yossi Azar and Debmalya Panigrahi, editors, *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pages 3392–3408. SIAM, 2025. doi:10.1137/1.9781611978322.111.

26 Noah G. Singer, Madhur Tulsiani, and Santhoshini Velusamy. Sketching approximations and lp approximations for finite csps are related, 2025. doi:10.48550/arXiv.2509.17926.

27 Luca Trevisan. Non-approximability results for optimization problems on bounded degree instances. In Jeffrey Scott Vitter, Paul G. Spirakis, and Mihalis Yannakakis, editors, *Proceedings on 33rd Annual ACM Symposium on Theory of Computing, July 6-8, 2001, Heraklion, Crete, Greece*, pages 453–461. ACM, 2001. doi:10.1145/380752.380839.

28 Luca Trevisan. Max cut and the smallest eigenvalue. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, pages 263–272. ACM, 2009. doi:10.1145/1536414.1536452.

29 Yuichi Yoshida. Optimal constant-time approximation algorithms and (unconditional) inapproximability results for every bounded-degree CSP. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, pages 665–674. Association for Computing Machinery, 2011. doi:10.1145/1993636.1993725.

Incremental Dominating Set

Ilan Doron Arad
Massachusetts Institute of Technology, Cambridge, MA, USA

Jonathan Gal
Technion – Israel Institute of Technology, Haifa, Israel

Joseph (Seffi) Naor
Technion – Israel Institute of Technology, Haifa, Israel

Abstract

Dominating Set is a fundamental problem in graph theory: given a graph, find a minimum-weight subset of vertices such that every vertex is either selected or adjacent to a selected vertex. In online settings where vertices arrive sequentially, comparing algorithms against an offline optimum with full knowledge of the input leads to extremely strong lower bounds, where even a simple star graph shows that any online algorithm must have competitive ratio $\Omega(\Delta)$, with Δ the largest degree of any vertex in the graph, matching the trivial strategy of selecting all vertices.

We study the *incremental* dominating set problem, where the optimal algorithm is constrained to the same choices available to online algorithms. This introduces a benchmark that enables a meaningful comparison between algorithms.

We present the first results for vertex-weighted graphs and randomized algorithms in this model. For incremental dominating set, we give an $O(\Delta)$-competitive deterministic algorithm and an $O(\log^2 \Delta)$-competitive randomized algorithm. We extend these results to the Connected Dominating Set problem using a linear-programming formulation that captures connectivity through local constraints. When the neighborhood of each arriving vertex is known *in advance*, deterministic algorithms achieve similar polylogarithmic competitive ratios as their randomized counterparts. Finally, we establish matching lower bounds, showing that our results are optimal up to constant factors.

2012 ACM Subject Classification Theory of computation → Online algorithms; Theory of computation → Graph algorithms analysis; Mathematics of computing → Graph theory

Keywords and phrases Algorithms, Online Algorithms, Dominating Set

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.5

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2606.27469

Funding *Joseph (Seffi) Naor*: Supported in part by ISF grant 3001/24 and United States – Israel BSF grant 2022418.

1 Introduction

Dominating Set is a fundamental and extensively studied problem in graph theory and algorithm design, e.g. [15, 18, 24]. Given a graph $G = (V, E)$ with vertex weights $w : V \to \mathbb{R}^+$, we say that vertex u dominates vertex v if either $u = v$ or $(u, v) \in E$. A subset of vertices $D \subseteq V$ is then called a dominating set if every vertex in V is dominated by at least one vertex in D. Variants include the Connected Dominating Set problem [10, 11, 27], which requires the dominating set to be connected, and the Total Dominating Set problem [2, 7], where every vertex must be dominated by a distinct neighbor. These, and related problems, have found applications in network routing [26], facility location [16, 22], efficient transmission [20], and more [23].


© Ilan Doron Arad, Jonathan Gal, and Joseph Naor;
licensed under Creative Commons License CC-BY 4.0

We consider online versions of these problems. In the online setting, domination requests arrive sequentially, requiring the algorithm to make irrevocable decisions, while maintaining feasibility at each step without knowledge of future requests [3, 8, 17].

Online domination has primarily been studied under two models: the known-graph and unknown-graph settings. In the known-graph model, the entire graph is known upfront, and requests for dominating specific vertices arrive sequentially. This model is a special case of the online set cover problem, for which the known competitive bounds are polylogarithmic in the number of sets and elements [1]. Several works have extended these bounds to variants of the known-graph model in the context of domination problems [21].

In the unknown-graph model, the vertices are revealed one by one along with the edges connecting them to previously seen vertices. Each revealed vertex must be immediately dominated. The version most commonly studied in this model is the late accept variant [3], in which a vertex can be added to the dominating set at any time after it arrives, but cannot be removed once it is added.

Most research in this setting analyzes online algorithms by comparing them to a powerful offline algorithm. Consider the star graph in Figure 1; it demonstrates that every online algorithm must have competitive ratio $\Theta(\Delta)$, where Δ is the maximum degree of a vertex, matching the trivial strategy that always adds all vertices to the dominating set. Thus, there has been very little research in this direction [3, 9].

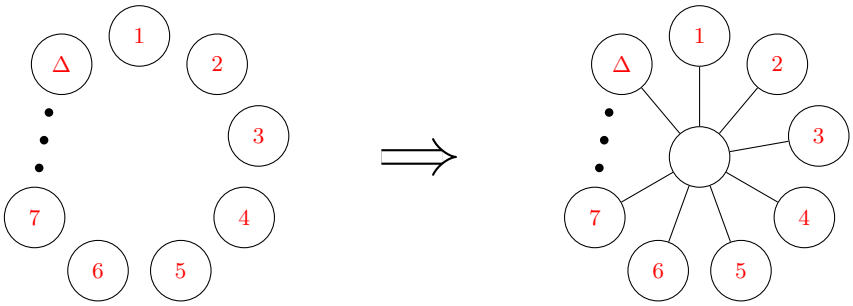


Figure 1 A star graph where the center vertex is revealed last. An online algorithm, unable to choose the center vertex before it is requested must select all leafs, while an offline algorithm needs only to select the center vertex, yielding a competitive ratio of $\Omega(\Delta)$. However, against the optimal **incremental** solution, which is also unable to choose the center vertex, the competitive ratio is 1.

To provide a more meaningful benchmark, we propose comparing online algorithms with their incremental counterparts. In the incremental dominating set (DS) problem, we are given a graph $G = (V, E)$ and an ordering $v_1, v_2, \ldots, v_n$ of the vertices. The goal is to choose a subset $S \subseteq V$ such that for every $t \in [n]$, the set $S \cap \{v_1, \ldots, v_t\}$ is a dominating set for the subgraph induced by $\{v_1, \ldots, v_t\}$. Since both the algorithm and the optimal incremental solution face the same arrival constraints, this benchmark is more informative. In the example in Figure 1, the competitive ratio is reduced to 1 from $\Omega(\Delta)$.

This problem naturally models constraints that online algorithms face in the unknown-graph setting. Boyar et al. [3] were the first to consider deterministic online algorithms for the incremental model on unweighted graphs in the unknown-graph setting. They showed that any deterministic online algorithm has a competitive ratio of $\Theta(\Delta)$. This result implies that, for unweighted graphs, no deterministic online algorithm can substantially outperform the naïve approach of simply selecting all vertices. However, the question of obtaining improved competitive bounds for other settings, such as weighted graphs, where achieving an $O(\Delta)$ competitive algorithm is non-trivial, or randomized algorithms, has remained open.

Table 1 Results for incremental DS and incremental CDS. Entries marked with - follow from results in other rows. All results apply to weighted graphs unless specified otherwise. Δ refers to the maximum degree of the graph induced by requested vertices, while Δ_G refers to the maximum degree of the revealed graph.

	Incremental DS	**Incremental CDS**	**Lower Bound**
Deterministic	$O(\Delta)$	$O(\Delta)$	$\Omega(\Delta)$ [3]
Randomized (Polynomial Time)	$O(\log^2 \Delta)$	$O(\log^2 \Delta)$	$\Omega(\log^2 \Delta)$
Randomized (General)			$\Omega(\log \Delta)$
Known Neighborhood Unweighted	$O(\log^2 \Delta_G)$	$O(\log^2 \Delta_G)$	$\Omega(\log^2 \Delta)$
Known Neighborhood (Polynomial Time)	$O(\log n \log \Delta)$	–	$\Omega(\log^2 \Delta)$
Known Neighborhood (Deterministic)			$\Omega(\log^2 \Delta / \log\log \Delta)$
Random Order Arrival	–	–	$\Omega(\log \Delta)$
Offline (Known Input)	$O(\log \Delta)$	$O(\log \Delta)$	$\Omega(\log \Delta)$

1.1 Our Contribution & Results

We address several open questions in online domination problems: obtaining competitive ratios for weighted graphs, randomized algorithms, and settings with additional structural information.

We start by formulating both incremental Dominating Set (DS) and incremental Connected Dominating Set (CDS) as covering problems. For incremental CDS we manage to encode both domination and connectivity requirements using a small number of linear constraints. This is the first use of such formulation in the online setting.

In the deterministic weighted setting, we show that incremental DS and incremental CDS admit an $O(\Delta)$-competitive algorithm, matching the unweighted lower bound [3]. For randomized algorithms, we achieve $O(\log^2 \Delta)$ competitive ratio and prove this is optimal up to a constant factor for polynomial-time algorithms. For computationally unbounded algorithms we prove an $\Omega(\log \Delta)$ lower bound.

We also study the **known neighborhood** model, where requesting a vertex reveals its full neighborhood. Surprisingly, knowing just the neighborhood of requested vertices in advance yields substantially stronger *deterministic* bounds: $O(\log^2 \Delta_G)$-competitive in the unweighted case for both incremental DS and incremental CDS where Δ_G is the maximum degree of the revealed graph (containing vertices which are not requested), and $O(\log n \log \Delta)$-competitive for weighted incremental DS. These competitive ratios hold at every step of the algorithm, and in particular the final solution remains competitive even if some revealed vertices were never requested. These bounds are asymptotically tight.

We also study offline and random-order models. In the offline setting, we establish tight $\Theta(\log \Delta)$ polynomial time approximation bounds unless NP $\subseteq$ BPP. In the random-order setting, we prove an $\Omega(\log \Delta)$ lower bound, showing that random arrival order does not substantially improve the competitive ratio.

A summary of our upper and lower bounds across different models appears in Table 1. We now discuss our results in detail.

1.1.1 Online Incremental

We begin with the online setting of incremental DS and CDS. Boyar et al. [3] have shown a $\Theta(\Delta)$ deterministic lower bound for incremental DS and incremental CDS, which matches the competitive bound of the trivial algorithm that selects all vertices in unweighted graphs. In contrast, we design a deterministic algorithm with a non-trivial competitive ratio for weighted graphs.

▶ **Theorem 1.** *There is a* $(\Delta + 1)$*-competitive deterministic algorithm for* incremental DS *and* incremental CDS*.*

In the randomized setting, the competitive ratio improves exponentially.

▶ **Theorem 2.** *There is an* $O(\log^2 \Delta)$*-competitive randomized algorithm for* incremental DS *and* incremental CDS*.*

We note that this result gives the first sublinear-competitive algorithm for weighted online CDS. In contrast, in the known-graph model (or in any model in which we only need to dominate a subset of the vertices) the problem reduces to non-metric TSP, for which obtaining an $f(n)$-approximation is hard for every polynomial-time computable function [11]. Thus, the incremental model allows substantially better guarantees.

We also study the **known neighborhood** model, where requesting a vertex v_i reveals its full neighborhood $N[v_i]$. Importantly, this extra information does not change the original setting, as the neighbors of v_i become visible to the algorithm, but cannot be added to the dominating set until they are requested. Surprisingly, this additional information enables deterministic algorithms to achieve polylogarithmic competitive ratios matching their randomized counterparts.

▶ **Theorem 3.** *When the neighborhood of requested vertices is known, and the graph is unweighted, then there is an* $O(\log^2 \Delta_G)$*-competitive deterministic algorithm for both* incremental DS *and* incremental CDS*.*

Here Δ_G denotes the maximum degree of the fully revealed graph, which may contain vertices that are never requested. The competitive ratio holds at every step of the algorithm, and remains valid even if the adversary stops requesting vertices before all revealed vertices have been requested. In the case where all revealed vertices are eventually requested we have $\Delta = \Delta_G$ and the algorithm is $O(\log^2 \Delta)$-competitive.

In the weighted case we still get polylogarithmic results for incremental DS.

▶ **Theorem 4.** *When the neighborhood of requested vertices is known, there is an* $O(\log n \log \Delta)$*-competitive deterministic algorithm for* incremental DS*.*

Here n denotes the number of revealed vertices and Δ the maximum degree of $G[V_n]$, the subgraph induced by requested vertices. As in the unweighted case, the competitive ratio holds at every step of the algorithm.

We establish matching lower bounds for our randomized algorithms.

▶ **Theorem 5.** *There is no polynomial-time online algorithm for* incremental DS *or* incremental CDS *with competitive ratio* $o(\log^2 \Delta)$ *unless* NP $\subseteq$ BPP*, even when the neighborhood of requested vertices is known.*

For deterministic algorithms, we obtain a stronger lower bound.

▶ **Theorem 6.** *There is no deterministic online algorithm for* incremental DS *or* incremental CDS *with competitive ratio* $o(\log^2 \Delta / \log\log \Delta)$*, even when the neighborhood of requested vertices is known.*

Finally, in the random-order setting where vertices arrive according to a uniformly random permutation, we establish a logarithmic lower bound.

▶ **Theorem 7.** *There is no online random-order algorithm for* incremental DS *or* incremental CDS *with competitive ratio* $o(\log \Delta)$*.*

The random-order model is a restriction of the fully adversarial model, the above lower bound immediately implies the following.

▶ **Corollary 8.** *There is no online algorithm for* incremental DS *or* incremental CDS *with competitive ratio* $o(\log \Delta)$*. This lower bound also holds when the neighborhood of requested vertices is known.*

1.1.2 The Offline Setting

We study the offline variants of incremental DS and CDS, where the entire graph and the vertex request order are known in advance. We show that these problems remain NP-complete and hard to approximate, and establish tight approximation bounds up to constant factors.

▶ **Theorem 9.** Incremental DS *and* incremental CDS *are* NP-complete*. Furthermore, unless* NP $\subseteq$ BPP *there is no polynomial-time* $o(\log \Delta)$ *approximation algorithm for either problem.*

Complementing this lower bound, we give a matching polynomial-time $O(\log \Delta)$-approximation algorithm for both problems.

▶ **Theorem 10.** *There is a polynomial-time* $O(\log \Delta)$*-approximation algorithm for* incremental DS *and* incremental CDS*.*

1.2 Our Techniques

We start in Section 3 by formulating both incremental DS and incremental CDS as covering problems. The formulation of incremental DS is standard, but for incremental CDS we introduce a more involved formulation that maintains connectivity throughout. For each newly requested vertex, we create different constraints depending on whether it connects to zero, one, or more than one existing connected component. This formulation captures both domination and connectivity requirements, while ensuring that both the maximum constraint size and maximum frequency are linear in Δ, enabling the application of standard techniques for online covering problems. This formulation is the foundation for most of our results.

As an immediate result from the linear formulation we prove the $O(\log^2 \Delta)$-competitive ratio for online randomized algorithms, and an $O(\log \Delta)$ approximation ratio for the offline setting, where both the graph and request order are known in advance. Another result of this formulation is a greedy deterministic $O(\Delta)$-competitive algorithm for the weighted variant of both settings.

In Section 4, we use the extra information provided by knowing the neighborhood of requested vertices to exponentially improve the competitive ratio for deterministic algorithms. When the graph is unweighted, we maintain both a fractional and an integral solution, coupled through a potential function that is exponential in the fractional coverage of elements not yet covered integrally. The exponential growth ensures that at most $O(|\mathrm{OPT}|)$ elements can be covered fractionally without being covered integrally. We combine this bound with an algorithm that guarantees that whenever a new uncovered vertex is requested, we add a

logarithmic number of vertices to the dominating set. Together, these techniques provide the desired competitive ratio. We note that our potential function approach is inspired by Alon et al. [1], yet our solution gives improved competitive bounds and handles a wider range of settings.

We note that Δ_G can increase over time, to handle this we use a guess-and-square technique that achieves the desired $O(\log^2 \Delta_G)$ competitive ratio, improving upon the $O(\log^3 \Delta_G)$ bound obtained by the standard guess-and-double approach (see Section 4.1.2 for more details). This technique is used several times throughout the paper for algorithms where the maximum degree of the graph is not known in advance.

To get a similar competitive ratio for incremental CDS, we partition constraints into two sets: the first one contains the constraints handled by the incremental DS algorithm and the second one contains at most $O(|\mathrm{OPT}|)$ other constraints. The small size of the second set ensures that satisfying these constraints preserves the competitive ratio.

For the weighted case, we augment the potential function with a term that bounds the difference between the cost of the integral and fractional solutions while simultaneously bounding the increase of the potential throughout the runtime, yielding a logarithmic competitive ratio for incremental DS.

To obtain matching lower bounds on our competitive ratios, we provide reductions from dominating set and online set cover. A key insight in both reductions is the use of auxiliary vertices that are revealed first and must be selected. Then, we reveal vertices which are covered by these auxiliary vertices, revealing information about the instance to the incremental algorithm (either the sets or the graph depending on the reduction), thus enabling us to construct the reductions. These reductions preserve both the instance size and the optimal solution size, allowing us to transfer known lower bounds to our setting.

For the random-order lower bound, we replicate vertices so that a random permutation induces the adversarial set cover sequence with high probability, transferring random-order set cover lower bounds to the random order setting.

2 Preliminaries

We introduce the notation used throughout the paper, formally defining the problems we study, and presenting their covering formulations.

Let $G = (V, E, w)$ denote an undirected graph with vertex set V, edge set E, and weight function $w : V \to \mathbb{R}^+$. Let $w(S) = \sum_{s \in S} w(s)$ denote the total weight of all elements in $S \subseteq V$. The set of all connected components of G is denoted by $C(G)$.

Without loss of generality, we assume that $V = \{v_1, \ldots, v_n\}$, and that vertices are requested in this order, i.e., vertex v_1 is requested first, then v_2, and so on. Denote $V_i = \{v_1, \ldots, v_i\}$ when the order is understood from context.

For a vertex v, we denote by $N(v)$ the set of neighbors of v and $N[v] = N(v) \cup \{v\}$. We define the natural extensions to sets $N[U] = \bigcup_{u \in U} N[u]$.

For any $H \subseteq V$, let $G[H]$ be the induced subgraph of G on H.

We now define the competitive ratio used to benchmark our algorithms throughout the paper.

▶ **Definition 11** (Competitive Ratio). *The* competitive ratio *of an online algorithm $\mathcal{A}$ is*

$$\sup \frac{w(\mathcal{A})}{w(\mathrm{OPT})},$$

where OPT *denotes the cost of the optimal algorithm, and the supremum is taken over all graphs, weight functions, and request sequences.*

In this paper, OPT always refers to an optimal **incremental** algorithm.

2.1 Problem Definitions

We begin by defining the dominating set and connected dominating set problems.

▶ **Definition 12** (Dominating Set). *For a graph G, a* dominating set *is a subset $S \subseteq V$ such that $\bigcup_{v \in S} N[v] = V$. The objective is to minimize $w(S)$.*

In addition to covering all vertices, we may impose the requirement that the dominating set be connected within each connected component of the graph.

▶ **Definition 13** (Connected Dominating Set). *A* connected dominating set (CDS) *is a* dominating set *of G such that for every connected component C of G, the set $S \cap C$ is connected.*

To model the decisions an online algorithm must make incrementally, we define the incremental dominating set problem and its connected variant. Recall that V_i denotes the set containing the first i vertices requested by the online process.

▶ **Definition 14** (Incremental Dominating Set). *For a graph G, an* incremental dominating set *is a subset $S \subseteq V$ such that for all $i \in [n]$, $S \cap V_i$ is a* dominating set *of $G[V_i]$.*

▶ **Definition 15** (Incremental Connected Dominating Set). *For a graph G, an* incremental connected dominating set *is a subset $S \subseteq V$ such that for all $i \in [n]$, $S \cap V_i$ is a* connected dominating set *of $G[V_i]$.*

In the online setting, the vertices of a graph are requested one by one. In this paper, we consider two information models: at step i, the algorithm either knows the induced subgraph $G[V_i]$, which corresponds to the standard vertex arrival model, or the neighborhood subgraph $G[N[V_i]]$. Throughout the paper, the competitive ratio is measured against OPT, the optimal incremental dominating set.

▶ **Definition 16** (Known Neighborhood Model). *In the* known neighborhood model*, at step i when vertex v_i is requested, the algorithm observes the induced subgraph $G[N[V_i]]$ rather than just $G[V_i]$. Vertices in $N[V_i] \setminus V_i$ are visible but cannot be selected for the dominating set until requested.*

We work in the late-accept model [3, 4], where vertices can be added to the solution at any time but cannot be removed once chosen. We now define the online variants of the incremental Dominating Set and incremental Connected Dominating Set problems.

▶ **Definition 17** (Online Incremental Dominating Set). *In the* online incremental dominating set *problem, the algorithm must maintain a dominating set of $G[V_i]$ at each step i.*

▶ **Definition 18** (Online Incremental Connected Dominating Set). *In the* online incremental connected dominating set *problem, the algorithm must maintain a connected dominating set of $G[V_i]$ at each step i.*

Since any superset of a CDS is also a CDS, the late accept setting does not violate the incrementality requirements of previous steps. Adding vertices to the solution at later steps preserves the CDS property for all previous graphs.

2.2 Covering Problems

In this paper, we use linear programs with covering constraints

▶ **Definition 19** (Covering Problem). *A* covering problem *is a linear program of the form:*

$$\begin{aligned} \min \quad & \sum_{i\in[n]} w(i)\cdot x_i \\ & s.t. \\ a^j \cdot \vec{x} &\geq 1, && \forall j \in [m] \\ x_i &\geq 0 && \forall i \in [n] \end{aligned} \tag{1}$$

where $a^j \in \{0,1\}^n$ *for all* $j \in [m]$.

3 Covering LPs for Incremental DS and Incremental CDS

We now formulate linear programming (LP) relaxations for incremental DS and incremental CDS, expressing both as covering problems with each constraint involving at most $O(\Delta)$ variables and each variable participating in $O(\Delta)$ constraints. Formulating both problems as covering LPs has two main advantages. First, it allows us to leverage techniques already known in the literature for online covering problems. Second, it provides a unified framework for designing algorithms for both problems simultaneously, an approach we exploit in later sections.

We formalize these observations in the following theorem,

▶ **Theorem 20.** *Both* incremental DS *and* incremental CDS *have* LP *relaxations that are covering problems. The size of each constraint is bounded, and so is the number of variables in each constraint. More precisely:*

1. *For* incremental DS*, each constraint involves at most* $\Delta + 1$ *variables.*
2. *For* incremental CDS*, each constraint involves at most* Δ *variables.*
3. *In both relaxations, each variable appears in at most* $\Delta + 1$ *covering constraints, and a single non-negativity constraint.*

3.1 Incremental DS as a Covering Problem

We begin by formulating the LP relaxation for incremental DS. For each vertex v_i, we introduce a variable $x_i \in [0,1]$.

Proof of Theorem 20(1). When v_i is requested, either v_i or one of its (previously requested) neighbors needs to be in the dominating set, so

$$\sum_{j:v_j\in N[v_i]\cap V_i} x_j \geq 1.$$

Due to the monotone nature of the solution, it suffices to ensure that each newly requested vertex is covered when it appears. This gives us the following relaxation

$$\begin{aligned} \min \quad & \sum_{i\in[n]} x_i \cdot w(v_i) \\ & \text{s.t.} \\ \sum_{j:v_j\in N[v_i]\cap V_i} & x_j \geq 1, && \forall i \in [n] \\ x_i &\geq 0, && \forall i \in [n] \end{aligned} \tag{2}$$

This relaxation is a covering problem. Each constraint involves at most $\Delta + 1$ variables since it includes v_i and its neighbors in V_i. Moreover, each variable x_j appears only in the non-negativity constraint and constraints for vertices v_i such that $v_j \in N[v_i]$. Since $|N[v_i]| \leq \Delta + 1$, each variable x_j appears in at most $\Delta + 1$ covering constraints. ◀

3.2 Incremental CDS as a Covering Problem

Here we prove Theorem 20(2) by identifying a small set of integral constraints that ensures the chosen subset is both a dominating set and is connected. We provide the intuition for the LP formulation here, and defer the details to the full version of the paper.

To express connectivity using local constraints, we classify each new vertex by how it interacts with existing components: it may start a new one, attach to one, or join several together. The cases require different types of constraints to maintain both domination and connectivity. To the best of our knowledge, this is the first time that connectivity is captured using only local constraints.

Proof of Theorem 20(2). We start by classifying the vertices into three types:

1. Opener – A vertex that, upon arrival, is not connected to any other vertex. Formally, $N[v_i] \cap V_i = \{v_i\}$. For example, the first vertex v_1 is always an opener.
2. Attacher – A vertex that, upon arrival, is connected to only one component. Formally, there exists $C \in C(G[V_{i-1}])$ such that for all $C' \neq C$, $N[v_i] \cap V_i \cap C' = \emptyset$.
3. Joiner – A vertex that connects at least two components upon arrival. Formally, there exist $C_1, C_2 \in C(G[V_{i-1}])$ such that $N[v_i] \cap V_i \cap C_1 \neq \emptyset$ and $N[v_i] \cap V_i \cap C_2 \neq \emptyset$.

We now add the appropriate constraints for each type to enforce both domination and connectivity.

Case 1: Opener. An Opener creates a new component and must be chosen to dominate itself. The only way to satisfy the dominating set requirement is to include it in the dominating set, thus $x_i \geq 1$.

Case 2: Attacher. An Attacher connects to one existing component C. The domination and connectivity constraints are satisfied as follows:

- If v_i is added to the CDS, at least one of its neighbors in C must also be in the CDS to maintain connectivity.
- If v_i is **not** added to the CDS, at least one of its neighbors must be added to satisfy the domination requirement.

Both scenarios are enforced by the constraint:

$$\sum_{v_j \in N(v_i) \cap V_i} x_j \geq 1.$$

This constraint is sufficient because the neighbors of v_i are already adjacent to the CDS, so adding any neighbor maintains both domination and connectivity constraints.

Case 3: Joiner. The joiner connects multiple components $C_1, \ldots, C_k$. After adding the joiner the dominating set of any pair of components must have a path between them, but such a path must go through the joiner. As a result the joiner must be added to the dominating set. To maintain connectivity:

- v_i must be included in the CDS, $x_i \geq 1$.
- For each component C_h, at least one neighbor of v_i within C_h must be included:

$$\sum_{v_j \in N(v_i) \cap V_i \cap C_h} x_j \geq 1.$$

This requirement is sufficient since any other vertex is already adjacent to the CDS, so adding a neighbor keeps the joiner connected to the dominating set in C_h.

The maximum size of a constraint is Δ for attachers. Moreover, each constraint corresponds to a vertex v_i and involves only its neighbors. Any variable x_j appears in one non-negativity constraint, at most one constraint for each neighbor v_i, and at most one constraint for v_j itself. Since each vertex has at most Δ neighbors, each variable appears in at most $\Delta + 2$ constraints overall. ◀

3.3 Algorithmic Implications

We start by showing the strength of Theorem 20 by combining it with several known theorems about covering problems to obtain immediate results.

The first of such results is for the offline problem, where from Srinivasan [25] we have the following theorem.

▶ **Theorem 21** (Srinivasan [25]). *For any covering problem with sparsity d (i.e., each variable appears in at most d constraints), there is an $O(\log d)$-approximation algorithm.*

Combining Theorems 20 and 21 yields the approximation ratio for the offline setting of both incremental DS and incremental CDS.

▶ **Theorem 10.** *There is a polynomial-time $O(\log \Delta)$-approximation algorithm for* incremental DS *and* incremental CDS*.*

Proof. By Theorem 20, both incremental DS and incremental CDS can be formulated as covering integer programs where each variable appears in at most $\Delta + 1$ constraints, meaning that the column sparsity is $d = \Delta + 1$. Applying Theorem 21 yields an $O(\log(\Delta + 1)) = O(\log \Delta)$-approximation algorithm for both problems. ◀

For the online setting, we apply the following theorem from Gupta and Nagarajan [13].

▶ **Theorem 22** (Gupta and Nagarajan [13]). *There exists an $O(\log k \cdot \log d)$-competitive randomized online algorithm for covering problems, where k is the maximum number of variables in any constraint and d is the sparsity (maximum number of constraints any variable appears in).*

Again, combining Theorems 20 and 22 immediately yields the randomized competitive ratio.

▶ **Theorem 2.** *There is an $O(\log^2 \Delta)$-competitive randomized algorithm for* incremental DS *and* incremental CDS*.*

Proof. By Theorem 20, the LP relaxations of both problems are online covering problems with row sparsity $k \leq \Delta + 1$ and column sparsity $d \leq \Delta + 1$. Applying Theorem 22 gives an $O(\log^2 \Delta)$-competitive randomized online algorithm. ◀

Greedy Deterministic Online Algorithm

We present a deterministic greedy algorithm for the weighted variants of incremental DS and incremental CDS. Upon the arrival of each covering constraint, the algorithm selects the minimum-weight vertex satisfying the constraint, adding it to the dominating set if it has not been selected previously. The analysis is based on the covering formulation of Theorem 20. Since each variable participates in at most $\Delta + 1$ covering constraints, the greedy algorithm is $(\Delta + 1)$-competitive. The complete proof is straightforward and is deferred to the full version of the paper.

▶ **Theorem 1.** *There is a $(\Delta + 1)$-competitive deterministic algorithm for* incremental DS *and* incremental CDS*.*

4 Know Thy Neighbor

In this section we present improved deterministic algorithms for the case where the neighborhood of each requested vertex v_i ($N[V_i]$) is revealed when v_i needs to be dominated. The setting remains incremental and vertices which have not been requested (yet were revealed earlier) cannot be used to cover requested vertices. In other words, when a vertex is requested, we learn about its neighbors, but only the requested vertex itself needs to be dominated.

Knowing the neighborhood of each requested vertex complies with the arrival model studied in previous sections, except that additional information is provided at runtime. This extra information allows us to design deterministic algorithms with polylogarithmic competitive ratios, a substantial improvement over the $\Omega(\Delta)$ lower bound from the setting where only requested vertices are known (without full neighborhood information).

Since revealed vertices need not all be requested, we distinguish between Δ_G, the maximum degree of the full revealed graph, and Δ, the maximum degree of $G[V_n]$, the subgraph induced by requested vertices.

In Section 4.1, we present an $O(\log^2 \Delta_G)$-competitive deterministic algorithm for the unweighted case, for both incremental DS and incremental CDS. Then, in Section 4.2, we show that in the weighted case there exists an $O(\log n \log \Delta)$-competitive deterministic algorithm for incremental DS. We leave improving the upper bound for weighted incremental CDS as an open problem.

This model differs from that of [14], where a vertex must be covered only when its last neighbor is revealed, and late acceptance is disallowed.

4.1 Unweighted Case

We first present a deterministic algorithm for incremental DS. Our approach is based on maintaining both a fractional solution and an integral dominating set D, coupled through a potential function. The potential function penalizes vertices that are covered fractionally but not integrally, ensuring that vertices with high fractional coverage are added to the integral solution.

The algorithm maintains a parameter m, initialized to an upper bound on the maximum degree of the revealed graph Δ_G, and updated dynamically to remain an upper bound as new vertices arrive (see Section 4.1.2). We also maintain a fractional solution x (possibly infeasible), initialized with $x_i = \frac{1}{4m}$ for each vertex v_i.

To track how well each vertex is covered fractionally, we define the potential cover c_v for each known vertex v. This measure has two components: the fractional coverage from revealed neighbors, and a correction term that accounts for potential neighbors that have not yet been revealed. Specifically, when a new neighbor of v is revealed, it contributes $\frac{1}{4m}$ to the fractional coverage, while simultaneously decreasing the correction term by the same amount, ensuring c_v remains unchanged. Formally, for each revealed vertex v we define:

$$c_v = \sum_{u_j \in N[v] \cap V_i} x_j + \frac{m - |N[v] \cap V_i|}{4m}.$$

Let D be the dominating set maintained by the algorithm at the current step, and denote the set of uncovered elements as $U_i = N[V_i] \setminus N[D]$. Define the potential at step i as $\Phi_i = \sum_{v \in U_i} m^{3c_v}$. The exponential term ensures that vertices with high fractional coverage cannot remain uncovered integrally.

At each step, we first initialize any newly revealed neighbors of v_i with fractional value $\frac{1}{4m}$. If v_i is uncovered, we snapshot the current potential as Φ_s, then increase the weights of vertices in $N[v_i] \cap V_i$ until v_i is covered fractionally. We then add the smallest subset of $N[v_i] \cap V_i$ to D such that the potential is restored to at most Φ_s; by Lemma 23 this subset has size $O(\log m)$. Finally, if v_i remains uncovered after this process, we add v_i itself to D. A pseudocode of the algorithm is described in Algorithm 1.

Algorithm 1 Unweighted Known Neighborhood Deterministic.

```
   // Notation as defined in the preceding text.
1  Initialize D ← ∅ (initial dominating set).
2  for vertex v_i ∈ V do
3      For each newly revealed vertex v_j, set x_j ← 1/(4m), and increment Φ_i by
       m^{3c_{v_j}} ≤ m^{3/4}. if v_i ∈ U_i then
4          Set Φ_s ← Φ_i.
5          while c_{v_i} < 1 do
6              For each v_j ∈ N[v_i] ∩ V_i, multiply x_j by two.
7          Add the smallest subset of vertices to D for which Φ_i ≤ Φ_s. // By Lemma 23,
           the size of such a subset is O(log m).
8      if v_i is not covered then
9          Add v_i to D.
```

To begin the analysis, we show that after each weight increase the algorithm can avoid an increase in the potential function by adding only a small subset of vertices to the dominating set in Line 7.

▶ **Lemma 23.** *If v_i is uncovered at step i, then there exists a subset of $N[v_i] \cap V_i$ of size at most $6 \log m$ such that adding the vertices in the subset to D ensures that the potential does not increase, i.e.,*

$$\Phi_i \leq \Phi_s,$$

where Φ_s is the potential prior to the weight increases in Line 6.

The proof is deferred to the full version of the paper. We next move from considering how much is added per update to bounding how often such updates happen.

▶ **Lemma 24.** *The total number of times the* **if** *statement in Line 3 in Algorithm 1 is triggered is at most $O(|\mathrm{OPT}| \log m)$.*

Proof. Every coordinate x_j is initialized to $1/(4m)$ and the algorithm only multiplies coordinates by 2 (Line 6). Moreover, Line 5 ensures that the algorithm never increases the value of any coordinate beyond 2. Thus, for every index j the number of times x_j can be doubled is at most $\log(8m) = O(\log m)$.

Since v_i must be dominated by a vertex in OPT, each time the **if** statement is triggered at least one coordinate corresponding to a vertex of OPT is doubled.

Combining these results, the total number of times that the **if** statement can be triggered is upper bounded by $O(|\mathrm{OPT}| \cdot \log m)$. ◀

Having bounded the number of update steps, we now show that the increase in the potential function itself remains bounded throughout the execution. Since each vertex added in the final while loop decreases the potential by a large amount, bounding the total potential increase immediately implies a bound on the number of vertices added in that phase.

▶ **Lemma 25.** *At step i, $\Phi_i \leq i \cdot \Delta_G \cdot m$.*

Proof. The potential can change in three places: when a vertex is requested, when weights are increased in Line 6, and when new vertices are revealed.

First, consider the moment when a vertex $v_i \in N[u]$ is requested. Its initial contribution of $1/(4m)$ is added to the first summand of c_u, but at the same time $|N[u] \cap V_i|$ increases by one, which decreases the correction term $(m - |N[u] \cap V_i|)/4m$ by exactly $1/(4m)$. The two changes cancel, and c_u does not change.

Second, by Lemma 23, after each weight increase in Line 6 the vertices added to D in Line 7 ensure that the potential does not increase, so weight increases cause no net increase in the potential.

Therefore the only source of increase is when new vertices are revealed. For each new vertex u, the increase in the potential is at most $m^{3c_u} \leq m$, since $c_u \leq \frac{1}{4}$ at reveal. After i steps, the number of revealed vertices is bound by $i \cdot \Delta_G$, the total increase is bounded by $i \cdot \Delta_G \cdot m$. ◀

We now use the bounded increase in Φ to bound the number of vertices added in the final loop of the algorithm. Intuitively, since each additional vertex in that phase reduces the potential by a large amount, a global bound on the total potential increase immediately limits how many such insertions can occur.

▶ **Lemma 26.** *Denote the number of total requested vertices throughout the algorithm as r. The number of vertices added by Lines 8-9 is bounded by $\frac{r \cdot \Delta_G}{m^2}$, and if $m \geq \Delta_G$ then the bound is $2 \cdot |\mathrm{OPT}|$.*

Proof. When we insert v_i to D in Line 9, we remove at least $m^{3c_u} \geq m^3$ from the potential. By lemma 25 the number of added vertices is at most

$$\frac{r \cdot \Delta_G \cdot m}{m^3} = \frac{r \cdot \Delta_G}{m^2}$$

If $m \geq \Delta_G$ then $|\mathrm{OPT}| \geq r/(\Delta_G + 1) \geq r/(2\Delta_G)$, giving us the upper bound. ◀

We are now ready to combine the previous bounds to obtain the overall competitive ratio.

▶ **Lemma 27.** *If $m \geq \Delta_G$, the competitive ratio of algorithm 1 is $O(\log^2 m)$.*

Proof. By Lemma 24 we enter the **if** in Line 3 at most $O(|\mathrm{OPT}| \cdot \log m)$ times, and by Lemma 23 each time we add $4 \log m$ vertices to D. By Lemma 26 we then add at most $2|\mathrm{OPT}|$ vertices in Lines 8-9. Hence

$$|D| \;\leq\; O(|\mathrm{OPT}| \cdot \log m) \cdot 4 \log m + 2|\mathrm{OPT}| \;=\; O(|\mathrm{OPT}| \cdot \log^2 m),$$

proving the claimed competitive ratio. ◀

We note that Algorithm 1 extends naturally to the case where constraints are of the form $N(v_i) \cap D$ (as with attacher constraints), where the requested vertex itself is excluded. If $N(v_i) = \emptyset$ then v_i is an isolated vertex and no constraint is enforced. Otherwise, all we need to do is replace all instances of $N[v_i]$ with $N(v_i)$. The potential function argument is unchanged, and the same $O(\log^2 \Delta_G)$-competitive bound holds.

4.1.1 Connected Dominating Set

In this part we extend the result of Section 4.1 to incremental CDS. The algorithm is similar to Algorithm 1, but with a special case for joiners (recall that a joiner is a vertex that joins two connected components, see Section 3.2 for the formal definition). When a joiner is requested, for each constraint created by it, we immediately add an arbitrary vertex from the constraint to the dominating set. We prove that the number of vertices added this way is bounded by $O(|\mathrm{OPT}|)$.

▶ **Lemma 28.** *The number of constraints added by joiners is bounded by* $2 \cdot |\mathrm{OPT}|$.

Proof. We actually prove a stronger statement, namely that the total number of constraints created by openers and joiners is at most $2 \cdot |\mathrm{OPT}|$. Let $\mathcal{O}$ be the number of openers and let $\mathcal{J}$ be the number of joiners. By the transformation in Section 3.2, every opener and joiner must belong to the dominating set, so $|\mathrm{OPT}| \geq \mathcal{O} + \mathcal{J}$.

We use a charging argument. When an opener is requested, it becomes the representative of its component. When a joiner is requested, each connected component it merges creates one constraint, which we charge to that component's representative. The joiner then becomes the representative of the merged component.

Each vertex can be charged at most twice: once when requested (as an opener or joiner), and at most once later as a representative. Therefore, the total number of constraints is at most $2(\mathcal{O} + \mathcal{J}) \leq 2 \cdot |\mathrm{OPT}|$. ◀

4.1.2 Guess and Increase

To extend our algorithm to the case where the maximum degree Δ_G is not known in advance, we use a guess-and-update technique inspired by the standard "guess-and-double" framework. A straightforward adaptation, starting with an underestimate of Δ_G and doubling it whenever the maximum degree exceeds the current guess, leads to a competitive ratio of

$$\sum_{j=1}^{\log \Delta_G} O(\log^2(\Delta_G/2^j)) \cdot \mathrm{OPT} = O(\log^3 \Delta_G) \cdot \mathrm{OPT},$$

which is asymptotically worse than the optimal bound. To handle this discrepancy, instead of doubling the current estimate of Δ_G, we square it upon discovering that the maximum degree violates the current guess.

We are now ready to prove the main theorem of the section.

▶ **Theorem 3.** *When the neighborhood of requested vertices is known, and the graph is unweighted, then there is an* $O(\log^2 \Delta_G)$*-competitive deterministic algorithm for both* incremental DS *and* incremental CDS*.*

Proof. Denote by OPT_i the cost of the optimal solution for the first i vertices, and let Δ_i be the maximum degree of the revealed graph at this step. Since each new vertex introduces additional constraints, it follows that $\mathrm{OPT}_i \leq \mathrm{OPT}$ for all i.

We maintain an adaptive upper bound Δ' on the current maximum degree, initialized to a constant. As vertices arrive, whenever $\Delta_i > \Delta'$, we update $\Delta' \leftarrow (\Delta')^2$, restart the algorithm with the new bound, and include in the global solution all vertices selected during the new run while retaining vertices selected during previous solutions.

By Lemma 27, each phase with bound Δ' has cost $O(\log^2 \Delta') \cdot \mathrm{OPT}$. Since Δ' grows quadratically, after k such increases the total cost is bounded by

$$O\left(\sum_{j=0}^{k} \log^2\left(\Delta_G^{1/2^{k-j}}\right)\right) \cdot \mathrm{OPT} \leq O\left(\sum_{j=0}^{\infty} \log^2\left(\Delta_G^{2^{-j}}\right)\right) \cdot \mathrm{OPT} = O(\log^2 \Delta_G) \cdot \mathrm{OPT}.$$

Therefore, the adaptive algorithm preserves the $O(\log^2 \Delta_G)$ competitive ratio even when Δ_G is unknown in advance.

To extend the result to incremental CDS, we use the CDS formulation from Section 3.2. We use Algorithm 1 to handle the attacher constraints (as defined in Section 3.2, these are the constraints corresponding to vertices that attach to an existing component) and achieve an $O(\log^2 \Delta_G)$-competitive dominating set for these constraints. Since the optimal solution for the attacher constraints is at most $|\mathrm{OPT}|$, the number of vertices selected by Algorithm 1 is bounded by $O(\log^2 \Delta_G) \cdot |\mathrm{OPT}|$. By Lemma 28, there are at most $2|\mathrm{OPT}|$ constraints created by openers and joiners, and adding one vertex per such constraint contributes at most $2|\mathrm{OPT}|$ additional vertices. Combining both contributions results in a valid incremental CDS of size $O(\log^2 \Delta_G) \cdot |\mathrm{OPT}|$, finishing the proof. ◀

4.2 Weighted Case

In this part, we show an $O(\log m \log \Delta)$ deterministic algorithm for incremental DS that uses an upper bound on the value of the optimal solution α, and a parameter m which represents an upper bound on the number of **requested** vertices n. In this section Δ represents the degree of $G[V_n]$, the subgraph induced by requested vertices, disregarding vertices which are revealed but not requested. We apply doubling to handle the case where α is not known, and we later show how to increase m dynamically, using a technique similar to doubling, to obtain the desired competitive ratio of $O(\log n \log \Delta)$. If the number of requested vertices n is known in advance we can set $m = n$ instead.

Our algorithm maintains both a fractional and an integral solution. We use the fractional solution to guide which vertices to add to the integral dominating set D, while a potential function ensures the two solutions remain balanced. To obtain the fractional solution, we apply the following result from Buchbinder et al. [5]:

▶ **Theorem 29.** *There is a deterministic $O(\log d)$-competitive fractional algorithm for covering problems with constraint size up to d.*

Since each covering constraint involves at most $\Delta + 1$ variables, Theorem 29 guarantees an online monotone fractional solution which is $O(\log \Delta)$ competitive.

Denote the dominating set as D. Let the set of vertices adjacent to requested vertices that are not yet dominated at step i be $U_i = N[V_i] \setminus N[D]$. Denote $c_{v_i} = \sum_{u_j \in N[v_i] \cap V_i} x_j$ as the fractional coverage of v_i, and define the potential at step i as

$$\Phi_i = \sum_{v \in U_i} m^{2c_v} + \exp\left(\frac{1}{2\alpha}\left(\sum_{v \in D} w(v) - 3 \log m \sum_{v_j \in V_i} w(v_j) x_j\right)\right).$$

The first sum ensures that every vertex is covered when it is requested, and the second term ensures that the integral and fractional solutions do not differ by too much.

At each step a vertex v_i is requested to be dominated. We update the potential function for the newly revealed neighbors of v_i, and then add the covering constraint $\sum_{j \in N[v_i] \cap V_i} x_j \geq 1$ to the LP. We increase the weights of vertices in $N[v_i] \cap V_i$ in the fractional solution until this constraint is satisfied. Then, we add a subset of vertices to D that ensures that the

potential does not increase. Vertices v_j with $w(v_j) > \alpha$ are kept at $x_j = 0$ in the fractional solution, ensuring that heavy vertices are never selected. The algorithm is described formally in Algorithm 2.

Algorithm 2 Known Neighborhood Deterministic.

```
   // Notation as defined in the preceding text.
1  Initialize D ← ∅ (initial dominating set).
2  Initialize x ← 0⃗, Φ_0 ← e^0 = 1.
3  for vertex v_i ∈ V do
4      For each newly revealed vertex u, increment Φ_i by m^{2c_u} = 1.
5      Add covering constraint for v_i and compute the online fractional solution x̃ from
        Theorem 29.
6      for j = 1, ..., i do
7          Set x_j ← x̃_j. // Update fractional solution
8          If Φ_i increased by the change in last line, add v_j to D
```

The analysis follows the same potential-function framework as in the unweighted case, where vertices are added to the integral solution in a way that ensures the potential function decreases. Combining this approach with the $O(\log \Delta)$-competitive fractional algorithm yields an $O(\log m \log \Delta)$ competitive ratio. The detailed analysis is deferred to the full version of the paper, and establishes the following result.

▶ **Theorem 4.** *When the neighborhood of requested vertices is known, there is an* $O(\log n \log \Delta)$*-competitive deterministic algorithm for* incremental DS.

5 Lower Bounds

5.1 Lower Bounds via Online Set Cover

We present a reduction from the online set cover problem to the online incremental Dominating Set problem. The reduction preserves the instance size up to polynomial factors and maintains the optimal solution size up to a constant additive term. This allows us to transfer known hardness results for online set cover and prove Theorems 5 and 6.

Given an instance of online set cover with universe $U = \{e_1, e_2, \ldots, e_n\}$ and collection $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$ with $S_i \subseteq U$, construct the incremental dominating set instance as follows. The vertex set is

$$V = \{r, t, t'\} \cup \mathcal{S} \cup U,$$

where r is a root adjacent to all set nodes, and t is a terminal adjacent to all element nodes, with a neighbor t'. The edge set is

$$E = (\{r\} \times \mathcal{S}) \cup \{(S_i, e_j) \mid e_j \in S_i\} \cup (U \times \{t\}) \cup \{(r, t), (t, t')\}.$$

That is, every set S_i is adjacent to r, a set S_i and element e_j are adjacent iff $e_j \in S_i$, every element of U is adjacent to t, and t is adjacent to both r and t'.

In the online adversarial setting, vertex r is requested first, followed by the vertices in $\mathcal{S}$. When an element is requested in the online set cover instance, the corresponding vertex is requested in the incremental DS instance. Once the set cover instance is finished, the terminal t is requested, followed by t', and then vertices in U which were not yet requested. A visualization of the reduction appears in Figure 2. We prove the following lemma,

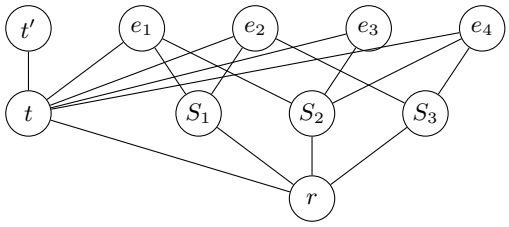


Figure 2 The resulting reduction from a set cover instance with 4 elements $E = \{e_1, e_2, e_3, e_4\}$ and 3 sets $S_1 = \{e_1, e_2\}$, $S_2 = \{e_1, e_3, e_4\}$ and $S_3 = \{e_2, e_4\}$. The vertices r and S_1, S_2, S_3 are requested first, and then when an element e_i is requested in the online set cover instance its corresponding vertex is requested in the reduction. After the online set cover instance is finished, t and t' are requested, followed by the rest of the vertices in E.

▶ **Lemma 30.** *There is an optimal solution for the reduced incremental instance that contains only elements belonging to* $\{r, t\} \cup \mathcal{S}$.

Proof. Let $\mathcal{I}$ be a solution to the reduced incremental instance. Vertex $r \in \mathcal{I}$, as r is the first requested vertex. Let $\mathcal{S}' = \mathcal{S} \cap \mathcal{I}$ and $U' = U \cap \mathcal{I}$. For each $u \in U'$ we choose some set $s_u \in \mathcal{S}$ that contains it. Define $\mathcal{S}_{U'} = \{s_u | u \in U'\}$, and let $\mathcal{I}' = \{r, t\} \cup \mathcal{S}' \cup \mathcal{S}_{U'}$.

The solution $\mathcal{I}'$ dominates all requested vertices: each $u \in U'$ is adjacent to $s_u \in \mathcal{I}'$, every set vertex is dominated by r, and r dominates itself. Since t' has only one neighbor t, any solution must contain t or t', so replacing whichever appears in $\mathcal{I}$ with t does not increase the solution size. Finally, $|\mathcal{S}_{U'}| \leq |U'|$ gives $|\mathcal{I}'| \leq |\mathcal{I}|$. ◀

We now proceed to proving that the reduction satisfies several properties.

▶ **Lemma 31.** *Our reduction satisfies the following properties:*

1. *The number of vertices in the resulting incremental instance is* $n + m + 3$.
2. *The optimal solution size in the incremental instance differs from the optimal set cover size by 2.*
3. *The reduction is online, meaning that each request in the online set cover instance is translated into a corresponding vertex in the online incremental DS instance.*
4. *There is an optimal solution that is connected.*

Proof. Property 1 is true by the way the reduction is defined.

The set cover solution selects a subset $\mathcal{S}' \subseteq \mathcal{S}$ to cover all the requested elements. The set $\{r, t\} \cup \mathcal{S}'$ is a valid solution to the reduction, since all vertices corresponding to sets in $\mathcal{S}$ are dominated by r, each requested element in U must be covered by an element in $\mathcal{S}'$, and all other elements in U are dominated by t. Thus a set-cover solution of size k yields an incremental solution of size $k+2$. By Lemma 30, let $\mathcal{I} \subseteq \{r, t\} \cup \mathcal{S}$ be the optimal solution to the reduced incremental instance, then $\mathcal{I} \backslash \{r, t\}$ is a feasible cover for the set cover instance of size $|\mathcal{I}| - 2$. Combining both directions proves the claim.

By Lemma 30, an optimal solution lies in $\{r, t\} \cup \mathcal{S}$. Since r and $\mathcal{S}$ are revealed before any element is requested, each subsequent element request corresponds exactly to the arrival of its matching vertex, and once t is requested it dominates all remaining elements of U without further additions, proving Property 3.

By Lemma 30, there exists an optimal solution containing only r, t, and elements from $\mathcal{S}$, where each vertex in $\mathcal{S}$ is adjacent to r. Since r is requested first and all other vertices in the optimal solution are connected to r, the partial solution remains connected at every step, thus proving Property 4. ◀

By combining these properties with known hardness results for online set cover [1, 19] the following results are established. The analysis is straightforward and is provided to the full version of the paper.

▶ **Theorem 5.** *There is no polynomial-time online algorithm for* incremental DS *or* incremental CDS *with competitive ratio* $o(\log^2 \Delta)$ *unless* NP $\subseteq$ BPP*, even when the neighborhood of requested vertices is known.*

▶ **Theorem 6.** *There is no deterministic online algorithm for* incremental DS *or* incremental CDS *with competitive ratio* $o(\log^2 \Delta / \log\log \Delta)$*, even when the neighborhood of requested vertices is known.*

5.2 Random Order Lower Bound

We present here a lower bound of $\Omega(\log n)$ on random order incremental DS. This proves Theorem 7. As a result, we can conclude that no algorithm has an $o(\log n)$ competitive ratio in the adversarial case, proving Theorem 8. We start with the result of Gupta et al. [12].

▶ **Theorem 32** (Theorem 5.1 in [12])**.** *There is an instance of set cover with n elements and $\Theta(n)$ sets, where any online algorithm for random order set cover is $\Omega(\log n)$-competitive.*

Let (S, E) be the set cover instance used in Theorem 32, where S is the collection of subsets and E is the ground set of elements. We construct an incremental Dominating Set instance as follows:

The vertex set includes all elements in E. For each subset $s_i \in S$, we create a clique $S_i = s_i^1, \ldots, s_i^{n^3}$. Each vertex in S_i is connected to all elements $e \in s_i$. Additionally, the clique $R = r^1, \ldots, r^{n^5}$ is added, where each $r^j \in R$ is connected to all vertices in every S_i.

Let η be the event that a vertex in R is requested first, and also for each S_i a vertex $s \in S_i$ is requested before all the vertices in E. We show that event η occurs with probability at least $1/2$.

▶ **Lemma 33.** *For sufficiently large n, $\mathbb{P}[\eta] \geq \frac{1}{2}$.*

The main idea is that $|R| = n^5$ is large relative to the $\Theta(n^4)$ remaining vertices, and each S_i's clique of size n^3 is large relative to the $\Theta(n)$ vertices of E that depend on it, so both events hold with high probability for sufficiently large n. The detailed calculation is provided in the full version of the paper. We now prove the lower bound.

▶ **Theorem 7.** *There is no online random-order algorithm for* incremental DS *or* incremental CDS *with competitive ratio* $o(\log \Delta)$.

Proof. Assume the event η occurs. Then some vertex in R is selected first, and for every $s_i \in S$, a vertex $s_i^j \in S_i$, which is already dominated by r, is requested before any $e \in E$.

When vertices in E are requested, each $e \in E$ is adjacent only to vertices s_i^j such that $e \in s_i$. To maintain the domination invariant, the algorithm must select such an s_i^j upon e's arrival. Selecting e itself would dominate only e, while selecting a neighboring s_i^j may also dominate future elements in E, so we may assume the algorithm avoids selecting e directly.

This simulates the random order set cover problem of Theorem 32, where the algorithm must cover elements of E using sets s_i. Given η, any algorithm incurs competitive ratio $\Omega(\log n)$. By Lemma 33 η occurs with constant probability, so the expected competitive ratio is $\Omega(\log n)$. The reduction produces a graph of size polynomial in n with maximum degree $\Delta = \Theta(n^5)$, so this establishes a lower bound of $\Omega(\log \Delta)$ for random-order incremental Dominating Set. ◀

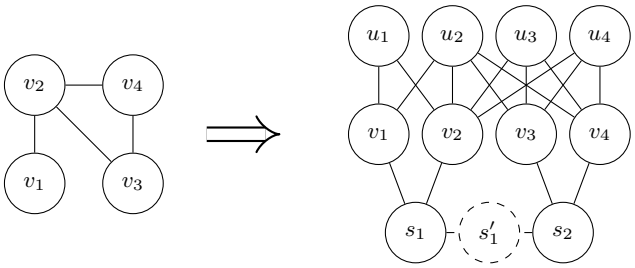


Figure 3 An example of the reduction presented in Section 5.3. The original graph is on the left, and the reduced instance on the right. Vertices in the reduced instance are requested from the bottom row to the top. The vertex s'_1 (dashed) is added for incremental CDS.

5.3 Hardness of Approximation

We construct a reduction from the offline Dominating Set (DS) problem to the incremental Dominating Set and incremental Connected Dominating Set (CDS) problems. The reduction preserves the number of vertices and maintains the maximum degree up to a constant factor. This construction is used to prove Theorem 9.

Let $G = (V, E)$ be an input graph with $V = \{v_1, \ldots, v_n\}$, maximum degree Δ, and minimum dominating set of size OPT. We define a new graph $G' = (V', E')$ by adding $S = \{s_1, \ldots, s_{n/\Delta}\}$ (wlog we assume that Δ divides n) and $U = \{u_1, \ldots, u_n\}$ to V. The full vertex set is $V' = V \cup S \cup U$, consisting of $n(2 + 1/\Delta)$ vertices.

We define the following edge sets in the new graph:

$$E_s = \{(s_i, v_{\Delta \cdot i+j}) \mid i \in [n/\Delta],\ j \in [\Delta]\},$$
$$E_u = \{(v_i, u_j) \mid (v_i, v_j) \in E\} \cup \{(v_i, u_i) \mid i \in [n]\}.$$

The final edge set is $E' = E_s \cup E_u$.

Intuitively, the set U is a duplicate of V as the edges are mirrored from V to U, and each vertex $v_i \in V$ is also connected to its counterpart $u_i \in U$. The vertices are requested in the following order: first the vertices in S, then those in V, and finally those in U.

The degree of vertex s_i is Δ, of v_i is bounded by $2\Delta + 1$ and of u_i is at most $\Delta + 1$. Thus, the maximum degree is $O(\Delta)$. In Figure 3 an example of the reduction is given.

We now prove that the size of any solution is almost preserved between the two models

▶ **Lemma 34.** *If D is a dominating set of G, then $D' = S \cup D$ is an incremental dominating set of G' with $|D'| \leq 3|D|$.*

Proof. If D is a dominating set of G, then the set $D' = S \cup D$ is an incremental dominating set in G'. Each vertex s_i is added to the solution immediately upon arrival. By construction, every vertex $v_i \in V$ is adjacent to at least one such s_j, ensuring that V is dominated once it arrives. For each vertex $u_i \in U$, if $v_i \in D$, then u_i is directly adjacent to $v_i \in D'$. Otherwise, since D is a dominating set in G, there exists a neighbor $v_j \in D$ of v_i, and in this case u_i is adjacent to $v_j \in D'$ via the edge $(v_j, u_i) \in E_u$.

The size of D' is

$$|D| + \frac{n}{\Delta} \leq |D| + 2\frac{n}{\Delta + 1} \leq 3|D|.$$

The first inequality holds because $\Delta > 0$, and the second follows from the standard lower bound on the size of any dominating set in a graph with maximum degree Δ. ◀

For the converse direction:

▶ **Lemma 35.** *If I is an incremental dominating set of G', then the set $D = \{v_i \mid v_i \in I \text{ or } u_i \in I\}$ is a dominating set of G and satisfies $|D| = O(|I|)$.*

Proof. Fix any $v_i \in V$.

- If $v_i \in D$, then v_i is trivially dominated.
- If $v_i \notin D$, but v_i is adjacent in G to some $v_j \in I$, then v_i is dominated via $v_j \in D$.
- Else, $u_i \in V'$ must be dominated in G', and since $u_i \notin I$, there must exist a vertex $v_j \in I$ such that $(v_j, u_i) \in E_u$. By construction of E_u, this implies $(v_i, v_j) \in E$, and $v_j \in D$.

In all cases, v_i is adjacent to some $v_j \in D$, so D is a dominating set of G.

To bound the size of D, note that $S \subseteq I$, and each $v_i \in D$ corresponds to at most two vertices in I. Therefore,

$$2|D| \geq |I| - |S| = |I| - \frac{n}{\Delta} \geq |I| - 2\frac{n}{\Delta + 1}.$$

Using $|D| \geq \frac{n}{\Delta+1}$, we get $4|D| \geq 2|D| + 2\frac{n}{\Delta+1} \geq |I|$. ◀

To extend the reduction to incremental CDS, we introduce an additional set of vertices $S' = \{s'_1, \ldots, s'_{n/\Delta-1}\}$ and add the edges (s_i, s'_i) and (s_i, s'_{i+1}) for all $i \in [n/\Delta - 1]$. These vertices are requested after S and must be selected to maintain connectivity. As a result, any valid incremental CDS solution must include the entire set S' to maintain connectivity. Moreover, the reduction preserves the asymptotic relationship between the size of the original dominating set and the size of the corresponding incremental CDS solution.

The reduction preserves the maximum degree up to a constant factor, and by Lemmas 34 and 35 the sizes of optimal solutions in G and G' are linearly related, a polynomial-time $o(\log \Delta)$-approximation algorithm for incremental DS would yield the same for classical Dominating Set, contradicting known hardness results [6]. This establishes the following.

▶ **Theorem 9.** Incremental DS *and* incremental CDS *are* NP-complete. *Furthermore, unless* NP $\subseteq$ BPP *there is no polynomial-time $o(\log \Delta)$ approximation algorithm for either problem.*

References

1 Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph (Seffi) Naor. The online set cover problem. *SIAM Journal on Computing*, 39(2):361–370, 2009. doi:10.1137/060661946.

2 Maryam Atapour and Nasrin Soltankhah. On total dominating sets in graphs. *International Journal of Contemporary Mathematical Sciences*, 4(6):253–257, 2009.

3 Joan Boyar, Stephan J Eidenbenz, Lene M Favrholdt, Michal Kotrbčík, and Kim S Larsen. Online dominating set. *Algorithmica*, 81(5):1938–1964, 2019. doi:10.1007/S00453-018-0519-1.

4 Joan Boyar, Lene M Favrholdt, Michal Kotrbčík, and Kim S Larsen. Relaxing the irrevocability requirement for online graph algorithms. *Algorithmica*, 84(7):1916–1951, 2022. doi:10.1007/S00453-022-00944-W.

5 Niv Buchbinder and Joseph (Seffi) Naor. The design of competitive online algorithms via a primal–dual approach. *Theoretical Computer Science*, 3(2-3):93–263, 2007.

6 Miroslav Chlebík and Janka Chlebíková. Approximation hardness of dominating set problems in bounded degree graphs. *Information and Computation*, 206(11):1264–1275, 2008. doi:10.1016/J.IC.2008.07.003.

7 Ernest J Cockayne, RM Dawes, and Stephen T Hedetniemi. Total domination in graphs. *Networks*, 10(3):211–219, 1980. doi:10.1002/NET.3230100304.

8 Bevan Das and Vaduvur Bharghavan. Routing in ad-hoc networks using minimum connected dominating sets. In *Proceedings of ICC'97-International Conference on Communications*, volume 1, pages 376–380. IEEE, 1997. doi:10.1109/ICC.1997.605303.

9 Minati De, Sambhav Khurana, and Satyam Singh. Online dominating set and independent set. *CoRR*, 2021. arXiv:2111.07812.

10 Ding-Zhu Du and Peng-Jun Wan. *Connected dominating set: theory and applications*, volume 77. Springer Science & Business Media, 2012.

11 Sudipto Guha and Samir Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20:374–387, 1998. doi:10.1007/PL00009201.

12 Anupam Gupta, Gregory Kehne, and Roie Levin. Random order online set cover is as easy as offline. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1253–1264. IEEE, 2022.

13 Anupam Gupta and Viswanath Nagarajan. Approximating sparse covering integer programs online. *Mathematics of Operations Research*, 39(4):998–1011, 2014. doi:10.1287/MOOR.2014.0652.

14 Hovhannes A. Harutyunyan, Denis Pankratov, and Jesse Racicot. Online Domination: The Value of Getting to Know All Your Neighbors. In *46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021)*, volume 202 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 57:1–57:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.MFCS.2021.57.

15 Stephen T Hedetniemi and Renu C Laskar. Bibliography on domination in graphs and some basic definitions of domination parameters. In *Annals of discrete mathematics*, volume 48, pages 257–277. Elsevier, 1991.

16 John N Hooker, Robert S Garfinkel, and CK Chen. Finite dominating sets for network location problems. *Operations Research*, 39(1):100–118, 1991. doi:10.1287/OPRE.39.1.100.

17 Koji M. Kobayashi. Improved Bounds for Online Dominating Sets of Trees. In *28th International Symposium on Algorithms and Computation (ISAAC 2017)*, volume 92 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 52:1–52:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ISAAC.2017.52.

18 D. (Dénes) König. *Theory of Finite and Infinite Graphs*. Birkhäuser, Boston, 1990. English translation of *Theorie der endlichen und unendlichen Graphen* (1950).

19 Simon Korman. On the use of randomization in the online set cover problem. *Weizmann Institute of Science*, 2:34, 2004.

20 Chung Laung Liu et al. *Introduction to combinatorial mathematics*, volume 181. McGraw-Hill New York, 1968.

21 Christine Markarian and Abdul-Nasser Kassar. Online deterministic algorithms for connected dominating set & set cover leasing problems. In *ICORES*, pages 121–128, 2020. doi:10.5220/0008866701210128.

22 Avraham Mehrez and Alan Stulman. The facility location problem when the underlying distribution is either dominated or dominating. *Zeitschrift für Operations Research*, 28:B157–B161, 1984. doi:10.1007/BF01920506.

23 EC Milner. The theory of graphs and its applications, 1964.

24 Oystein Ore. *Theory of graphs*. Providence, R.I. : American Mathematical Society, 1962.

25 Aravind Srinivasan. Improved approximation guarantees for packing and covering integer programs. *SIAM Journal on Computing*, 29(2):648–670, 1999. doi:10.1137/S0097539796314240.

26 Jie Wu and Hailan Li. A dominating-set-based routing scheme in ad hoc wireless networks. *Telecommunication Systems*, 18:13–36, 2001. doi:10.1023/A:1016783217662.

27 Jiguo Yu, Nannan Wang, Guanghui Wang, and Dongxiao Yu. Connected dominating sets in wireless ad hoc and sensor networks–a comprehensive survey. *Computer Communications*, 36(2):121–134, 2013. doi:10.1016/J.COMCOM.2012.10.005.

Bichromatic Geometric Spanners

Theodore Fung
Department of Mathematics, California State University Northridge, Los Angeles, CA, USA

Csaba D. Tóth
Department of Mathematics, California State University Northridge, Los Angeles, CA, USA
Department of Computer Science, Tufts University, Medford, MA, USA

Abstract

For an edge-weighted graph $G = (V, E)$ and a stretch parameter $t \geq 1$, a t-spanner is a subgraph $H \subseteq G$ such that the shortest path distances in G and H satisfy $\delta_H(u, v) \leq t\,\delta_G(u, v)$ for all $u, v \in V$. In metric spanners, V is a finite metric space, and G is the complete graph with edge weights corresponding to the distances between the endpoints. When G is the complete graph on n points in the plane, $O(n)$-size t-spanners are possible for any $t > 1$: For every $\varepsilon > 0$, there is an $(1+\varepsilon)$-spanner with $O(n/\varepsilon)$ edges (the stretch can be arbitrarily close to 1).

When $G = K(R, B)$ is the complete bipartite graph on n bichromatic points in the plane, in general, no spanner construction can achieve stretch $t < 3$ with $o(n^2)$ edges. Bose et al. (SICOMP 2009) constructed a $(3+\varepsilon)$-spanner with $O(n \log n)$ edges for any constant $\varepsilon > 0$. Our main result is a new construction for a $(3+\varepsilon)$-spanner with $O(\sqrt{1/\varepsilon} \cdot n)$ edges. Eliminating the $O(\log n)$ factor resolves a problem left open for more than 17 years, and raises a new research problem about optimizing the dependence on ε. We also study spanners for $G = K(R, B)$ on n bichromatic points on the real line: In this case, we show that the MST of $K(R, B)$ is a 7-spanner, and we construct a 3-spanner with at most $2n - 3$ edges.

2012 ACM Subject Classification Mathematics of computing → Paths and connectivity problems; Theory of computation → Sparsification and spanners; Theory of computation → Approximation algorithms analysis

Keywords and phrases Euclidean spanner, bichromatic points, computational geometry

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.6

Category APPROX

Related Version *Full Version*: `https://arxiv.org/abs/2607.10062`

Funding Research supported in part by the NSF award DMS-2154347.

Acknowledgements We are grateful to Sujoy Bhore for initial conversions on the structure of bichromatic spanners on the real line.

1 Introduction

A t-spanner for an edge-weighted graph $G = (S, E)$ is a subgraph $H = (S, E')$ that contains, for every edge $ab \in E$, an ab-path of weight at most $t \cdot w(ab)$. Let S be a set of n bichromatic points in Euclidean plane, with coloring $c : S \to \{\text{red}, \text{blue}\}$. Let R and B, resp., denote the set of red and blue points in S, and let $K(R, B)$ denote the complete bipartite graph on the partite sets R and B. The weight of an edge ab is the Euclidean length $|ab|$ of the line segment ab.

For a set S of n (uncolored) points in the plane and any $\varepsilon > 0$, there exists a $(1+\varepsilon)$-spanner with $O(n/\varepsilon)$ edges [18, 23]. Bose et al. [15] noted that for a complete bipartite graph $K(R, B)$, in general, one cannot find a t-spanner with $o(n^2)$ edges and stretch $t < 3$. Specifically, for every $\varepsilon > 0$, one can arrange $\lceil n/2 \rceil$ red and $\lfloor n/2 \rfloor$ blue points in two clusters of diameter $\frac{\varepsilon}{3}$ at unit distance apart: If a subgraph $H \subseteq K(R, B)$ misses any bichromatic

© Theodore Fung and Csaba D. Tóth;
licensed under Creative Commons License CC-BY 4.0

edge ab, then every ab-path in H has length at least $(3-\varepsilon)|ab|$. On the positive side, Bose et al. [15] constructed, for every bichromatic set $S = R \cup B \subset \mathbb{R}^2$ and constant $\varepsilon > 0$, an $(5+\varepsilon)$-spanner with $O(n)$ edges and a $(3+\varepsilon)$-spanner with $O(n \log n)$ edges. They did not analyze the dependency on the stretch parameter ε. The main problem left open in their work is whether a bichromatic $(3+\varepsilon)$-spanner with $O(n)$ edges always exists. Our main result settles this problem in the affirmative.

▶ **Theorem 1.** *For every $\varepsilon > 0$ and every set of n bichromatic points in the plane, there exists a $(3+\varepsilon)$-spanner with $O(\sqrt{1/\varepsilon} \cdot n)$ edges, which can be constructed in $O(\sqrt{1/\varepsilon} \cdot n \log n)$ time.*

While we settle the problem posed by Bose et al. [15], our result raises a new problem to determine the optimal tradeoff between $\varepsilon > 0$ and the size of a bichromatic $(3+\varepsilon)$-spanner in the plane. Theorem 1 generalizes to Euclidean d-space for constant dimension $d \geq 2$, and yields a $(3+\varepsilon)$-spanner with $O(n/\varepsilon^{(d-1)/2})$ edges. Moreover, in the related version of the paper, Theorem 1 has been generalized to every set of n multichromatic points in the plane. We describe the spanner construction and analyze it in detail in the plane, and then briefly sketch the generalization to higher dimensions.

We also study spanners for $K(R, B)$ in the real line. The lower bound construction by Bose et al. [15] can be realized already in the line: It shows that, in general, a spanner construction cannot achieve stretch $t < 3$ with $o(n^2)$ edges. We construct a 3-spanner with at most $2n - 3$ edges (Proposition 9). We also show that the bichromatic Euclidean MST (i.e., the MST of the complete bipartite graph $K(R, B)$ with Euclidean edge weights) is a 7-spanners; and the stretch bound 7 is the best possible (Proposition 10). For comparison, recall that for n (uncolored) points on the line, the Euclidean MST is an x-monotone path, which is a 1-spanner. However, for n (uncolored) points in the plane, the Euclidean MST has unbounded stretch.

Comparison to Previous Work and Technical Highlights. As mentioned above, Bose et al. [15] constructed a $(3+\varepsilon)$-spanner with $O(n \log n)$ edges for a bichromatic set $S = R \cup B$ of n points and constant $\varepsilon > 0$. Their spanner construction uses a hierarchical space partition, but a crucial step of their approach has a $\Omega(n \log n)$ barrier. Our approach is significantly different and perhaps conceptually simpler. We briefly review the spanner construction in [15] and point out key differences from our construction.

For a bichromatic set $S = R \cup B$ of n points in the plane, Bose et al. [15] use the classical *well-separated pair decomposition* (WSPD, for short) by Callahan and Kosaraju [17]. A WSPD with separation parameter $\varrho > 0$ is a collection of unordered pairs $\mathcal{P} = \big\{\{S_i, T_i\} : i = 1, \ldots, m\big\}$ such that every point pair $\{s, t\} \in \binom{S}{2}$ appears there is an i such that $|\{s, t\} \cap S_i| = 1$ and $|\{s, t\} \cap T_i| = 1$; and $\operatorname{dist}(R_i, B_i) \geq \varrho \cdot \max\{\operatorname{diam}(R_i), \operatorname{diam}(B_i)\}$ for all i. A WSPD of size $m = O(n)$ can be construed using split trees [17, 26] or compressed quadtrees [20]. For an uncolored set of n points in $\mathbb{R}^d$, WSPDs with separation parameter $\varrho = \Theta(1/\varepsilon)$ were previously used for constructing $(1+\varepsilon)$-spanners of size $O(n/\varepsilon^d)$ [17, 19, 21]; and this bound is the best possible. Given a WSPD, each set S_i (resp., T_i) could be either monochromatic or bichromatic. Bose et al. [15] choose an arbitrary *representative* from each monochromatic set S_i and T_i; and two representatives of different colors from bichromatic sets. Their spanner consists of specific edges chosen for each pair $\{S_i, T_i\}$ of the WSPD. If S_i and T_i are bichromatic, they add a single edge between a pair of representative from S_i and T_i, respectively. However, if a set S_i is monochromatic, they need a path from all points of S_i to its representative: They achieve this by connecting all points in S_i to the

representative of a closest cluster $S_j = \mathrm{cl}(S_i)$ that contains a point of the other color (hence every point in S_i will have a 2-hop path to the representative of S_i). They prove that this graph is $(5+\varepsilon)$-spanner of size $O(n)$. They further show that *if* the WSPD has the additional property that $\min\{|S_i|, |T_i|\} = 1$ for all i, then their construction yields a $(3+\varepsilon)$-spanner. However, a WSPD with the additional property that size $m = O(n \log n)$, which is the best possible [17, 26], and dominates the size of their $(3+\varepsilon)$-spanner.

Our spanner design is crucially different in handling well-separated pairs with monochromatic clusters. If both S_i and T_i are bichromatic, we also add just two edges between representative. If S_i is monochromatic, we connect all points in S_i to the representative of T_i of the opposite color (and similarly, if T_i is monchromatic, we connect all points in T_i to a representative of S_i). Overall, for each well-separated pair (S_i, T_i), a star, a bistar, or just a pair of edges between S_i and T_i (see Cases 1–3 in Section 2.2). In particular, we do not use a detour via any closest cluster $S_j = \mathrm{cl}(S_i)$. This approach would already give a bichromatic $(3+\varepsilon)$-spanner of size $O(n)$ for any constant $\varepsilon > 0$.

Instead of a quadtree-based WSPD, however, we use a custom-made space partition scheme with rectangular cells of aspect ratio $\Theta(\sqrt{1/\varepsilon})$ to improve the dependence on $\varepsilon > 0$. In a standard quadtree-based WSPD with separation parameter $\varrho = \Theta(1/\varepsilon)$, the number of pairs is $m = O(\varrho^2 n) = O(n/\varepsilon^2)$ [20, Theorem 4.1.3]. Even though the separation parameter must be $\varrho = \Theta(1/\varepsilon)$ for a $(1+\varepsilon)$-spanner on uncolored point sets, our stretch analysis for a $(3+\varepsilon)$-spanner also works with a smaller separation parameter $\varrho = \Theta(\sqrt{1/\varepsilon})$. This would reduce the size of our spanner to $O(n/\varepsilon)$.

Furthermore, we have noticed that our inductive stretch analysis goes through even if we replace the square cells of a quadtree with rectangular cells with a larger aspect ratio provided that any two edges between the well-separated cells make an angle $O(\sqrt{\varepsilon})$. This relaxes the separation condition to $\mathrm{dist}(S_i, T_i) \geq \Theta(\sqrt{1/\varepsilon}) \cdot \max\{\mathrm{width}(S_i), \mathrm{width}(T_i)\}$, and further reduces the size of our $(3+\varepsilon)$-spanner to $O(n/\sqrt{\varepsilon})$. We present this construction (without standard quadtrees and WSPDs) in Section 2. Our system of tilings with exponentially increasing scales and $\Theta(\sqrt{\varepsilon})$ directions is presented in Section 2.1, and our spanner is constructed based on pairs of well-separated tiles in Section 2.2. The stretch and sparsity analyses are presented in Section 2.3 and Section 2.4, respectively. We analyze the running time of our spanner construction in Section 2.5, and briefly outline generalizations to dimensions $d \geq 3$ in Section 2.6.

Previous work on bichromatic points. Bose et al. [16] studied a related problem: Given a finite set S in the plane, partition S into two nonempty color classes, $S = R \cup B$, so as to minimize the spanning ratio of the resulting complete bipartite graph $G = K(R, B)$, where the weight of every edge equals its Euclidean length. This problem is fundamentally different: In our problem, the color classes and $K(R, B)$ are fixed, and we need to find a small spanner in $K(R, B)$.

Bereg et al. [8] recently considered Euclidean spanners for the graph $G(R, B)$ of all red-blue and blue-blue edges (excluding red-red edges) on a set $S = R \cup B$ of n bichromatic points in $\mathbb{R}^d$. They constructed a 1.998-spanner of size at most $13n$ for $G(R, B)$ in the plane, and showed that the stretch of any spanner for $G(R, B)$ with $O(n)$ edges is at least $1 + \eta$ for some absolute constant $\eta > 0$. They also constructed a 1-spanner of size at most $2n$ for points in the real line.

Bichromatic variants for network optimization problems on n points in the plane, restricted to $K(R, B)$, have been studied extensively. In general, these problems are known to be more challenging than their original uncolored versions: For example, the Euclidean MST is always

a noncrossing straight-line graph; however, the bichromatic Euclidean MST may have crossing edges [3, 12]. Researchers studied bichromatic MSTs [11], bichromatic noncrossing MSTs [4, 14], bichromatic bottleneck MSTs [2], and bicromatic bottlenck noncrossing matchings [13]. Bichromatic TSP [22, 27], Euclidean TSP [6] and bichromatic Voronoi diagrams [5, 25] were also studied. Many of these problems are already nontrivial for bichromatic points on a line [1, 7].

2 Bichromatic $(3+\varepsilon)$-Spanners in the Plane

2.1 Tilings in the Plane

We begin by establishing some preliminary definitions. For a given $\varepsilon \in (0,1)$, let $\delta = \sqrt{\varepsilon/3} = \Theta(\sqrt{\varepsilon})$. We construct a tiling of the plane with rectangular tiles of aspect ratio $1/\delta$.

▶ **Definition 2.** *For every $i, j \in \mathbb{Z}$, we define the* tile

$$T(i,j) = [i-1, i] \times [(j-1)\delta, j\delta].$$

The set of all tiles forms a tiling $T = \{T(i,j) : i, j \in \mathbb{Z}\}$.

In our spanner construction, we use several affine copies of the tiling T, each copy is obtained by transforming all tiles in T by the same affine transformation (rotation, scaling, and translation). Recall that every tile $T(i,j) \in T$ is an axis-aligned rectangle of width 1. Let $T' = \{T(i,j) + (0, \frac{\delta}{2}) : i, j \in \mathbb{Z}\}$ denote the tiling obtained by T by translation, with vector $(0, \frac{\delta}{2})$. For every angle $\theta \in [0, 2\pi)$ and every $\lambda > 0$, let $T_{\theta,\lambda}$ (resp., $T'_{\theta,\lambda}$) be the tiling obtained from T (resp., T') by a counterclockwise rotation through θ about the origin, followed by a dilation with factor λ.

However, we restrict ourselves to a discrete family of tilings: We use only $O(\sqrt{1/\varepsilon})$ angles θ, and a discrete set of exponentially increasing scaling factors, that we define now. Let $\varphi = \arctan(\delta/22)$ and $k = \lceil \pi/\varphi \rceil$. Note that $\varphi \in \Theta(\sqrt{\varepsilon})$ and $k \in \Theta(\sqrt{1/\varepsilon})$. Our discrete set of k angles is $\Phi = \{2r\pi/k : r = 0, 1, \ldots, k-1\}$. Let $\gamma = 3^{1/11}$, and note that $3^{1/11} < 10/9$. Our discrete set of scaling factors is $\Lambda = \{\gamma^p : p \in \mathbb{Z}\}$. The family of all tilings with rotation angle in Φ and scaling factor in Λ is

$$\mathcal{T} = \bigcup_{\theta \in \Phi} \bigcup_{\lambda \in \Lambda} (T_{\theta,\lambda} \cup T'_{\theta,\lambda}).$$

The following lemma provides an important property of the family of tilings $\mathcal{T}$, which also plays a crucial role in the stretch analysis of our construction.

▶ **Lemma 3.** *For every pair of distinct points $a, b \in \mathbb{R}^2$, there exist $\theta^* \in \Phi$, $\lambda^* \in \Lambda$ and $i^*, j^* \in \mathbb{Z}$ such that at least one of the following holds:*

1. *tiles $T_{\theta^*,\lambda^*}(i^*, j^*)$ and $T_{\theta^*,\lambda^*}(i^*+10, j^*)$ each contain one of a and b,*
2. *tiles $T'_{\theta^*,\lambda^*}(i^*, j^*)$ and $T'_{\theta^*,\lambda^*}(i^*+10, j^*)$ each contain one of a and b.*

Proof. Let ψ be the counterclockwise angle between the positive x-axis and vector $\vec{ab}$. We choose $\theta^* \in \Phi$ such that θ^* is the closest angle to ψ; i.e., θ^* is the solution to the minimization problem

$$\theta^* = \arg\min_{\theta \in \Phi} |\theta - \psi|.$$

We may assume (by suitable rotation and reflections, if necessary) that $\theta^* = 0$, $\varphi \geq 0$, and $x(a) < x(b)$, where $x(a)$ and $x(b)$, resp., denote the x-coordinate of a and b. Next, we show that there exists a scale $\lambda^* \in \Lambda$ that satisfies

$$9\lambda^* \leq x(b) - x(a) < 10\lambda^*. \tag{1}$$

Let $\lambda^* = \gamma^{p^*}$, where

$$p^* = \left\lfloor \frac{\ln \frac{x(b)-x(a)}{9}}{\ln \gamma} \right\rfloor.$$

This gives $\gamma^{p^*} \leq \frac{x(b)-x(a)}{9} \leq \gamma^{p^*+1}$; and combined with $\gamma = 3^{1/11} < 10/9$, it implies Equation (1).

Since T_{θ^*,λ^*} is a tiling of $\mathbb{R}^2$, there exist $i^*, j^* \in \mathbb{Z}$ such that $a \in T_{\theta^*,\lambda^*}(i^*, j^*)$. Now Equation (1) implies that $x(a) + 9\lambda^* \leq x(b) \leq x(a) + 10\lambda^*$, and so $b \in T_{\theta^*,\lambda^*}(i^* + 10, j^{**})$ for some $j^{**} \in \mathbb{Z}$. Let $y(a)$ and $y(b)$, resp., denote the y-coordinate of a and b. If $y(a) = y(b)$, then we must have $j^* = j^{**}$. Suppose $y(a) \neq y(b)$, we may assume w.l.o.g. that $y(a) < y(b)$ and $\lambda^* = 1$. Under these assumptions, we have $T_{\theta^*,\lambda^*}(i^*, j^*) = T(i^*, j^*)$, which is an axis-aligned rectangle of width 1 and height δ. By our choice of φ, we can bound $y(b)$ from above, and conclude that

$$y(b) - y(a) = (x(b) - x(a)) \tan(\psi) \leq 11 \tan(\varphi) = \frac{\delta}{2}. \tag{2}$$

A horizontal median of the tile $T_{\theta^*,\lambda^*}(i^*, j^*)$ subdivides $T_{\theta^*,\lambda^*}(i^*, j^*)$ into two congruent rectangles, that we call the *upper* and *lower half* of $T_{\theta^*,\lambda^*}(i^*, j^*)$. Then one of the following two cases occurs

- **Case 1:** a is in the lower half of $T_{\theta^*,\lambda^*}(i^*, j^*)$. In this case, we have $0 \leq y(a) \leq \delta/2$, and Equation (2) gives $0 \leq y(b) \leq \delta/2$. Thus $a \in T_{\theta^*,\lambda^*}(i^*, j^*)$ and $b \in T_{\theta^*,\lambda^*}(i^* + 10, j^*)$.
- **Case 2:** a is in the upper half of $T_{\theta^*,\lambda^*}(i^*, j^*)$. In this case, we have $\delta/2 \leq y(a) \leq \delta$, and Equation (2) gives $\delta/2 \leq y(b) \leq \frac{3}{2}\delta$. Consequently, $a \in T'_{\theta^*,\lambda^*}(i^*, j^*)$ and $b \in T'_{\theta^*,\lambda^*}(i^* + 10, j^*)$.

This concludes the proof of Lemma 3. ◀

2.2 Spanner Construction

We are given a set $S = R \cup B$ of n bichromatic points in the plane and a parameter $\varepsilon > 0$. Let $\mathcal{T}$ be the family of tilings defined in Section 2.1. For every nonempty tile $T_{\theta,\lambda}(i, j)$, we denote by $N_{\theta,\lambda}(i, j)$ its *neighborhood* obtained by taking the union of $T_{\theta,\lambda}(i, j)$ and its eight surrounding tiles at the same scale. We say that $N_{\theta,\lambda}(i, j)$ is *bichromatic* if it contains at least one point of each color. Similarly, $N_{\theta,\lambda}(i, j)$ is *monochromatic* if all points in $N_{\theta,\lambda}(i, j)$ have the same color. It is important to note that $N_{\theta,\lambda}(i, j)$ can be bichromatic even if all points in $T_{\theta,\lambda}(i, j)$ are of the same color. For our construction, we begin by initializing a graph $G = (S, E)$ to be the empty graph.

For a set $Q \subseteq S$, we say that point q is *leftmost* in $Q \cap N_{\theta,\lambda}(i, j)$ if, after a clockwise rotation of $Q \cap N_{\theta,\lambda}(i, j)$ through θ, its x-coordinate is minimal. Likewise, the point q is *rightmost* in $Q \cap N_{\theta,\lambda}(i, j)$ if, after the same rotation, its x-coordinate is maximal. Note that several leftmost (resp., rightmost) points may exist (which have the same x-coordinate after suitable rotation). We choose a rightmost red point r^* in $N_{\theta,\lambda}(i, j)$ and a leftmost blue point b^* in $N_{\theta,\lambda}(i + 10, j)$. Similarly, we choose a rightmost blue point b^{**} in $N_{\theta,\lambda}(i, j)$ and a leftmost red point r^{**} in $N_{\theta,\lambda}(i + 10, j)$.

Consider every $\theta \in \Phi$, $\lambda \in \Lambda$, and $i, j \in \mathbb{Z}$ such that both $T_{\theta,\lambda}(i,j)$ and $T_{\theta,\lambda}(i+10,j)$ are nonempty (the translated tiles are handled similarly if both $T'_{\theta,\lambda}(i,j)$ and $T'_{\theta,\lambda}(i+10,j)$ are nonempty). We distinguish between four cases based on whether the neighborhoods $N_{\theta,\lambda}(i,j)$ and $N_{\theta,\lambda}(i+10,j)$ are monochromatic or bichromatic:

- **Case 0:** Both $N_{\theta,\lambda}(i,j)$ and $N_{\theta,\lambda}(i+10,j)$ are monochromatic, and all points in $N_{\theta,\lambda}(i,j) \cup N_{\theta,\lambda}(i+10,j)$ have the same color. In this case, we do not add any edges G.
- **Case 1:** Both $N_{\theta,\lambda}(i,j)$ and $N_{\theta,\lambda}(i+10,j)$ are monochromatic, but they contain different colors. Assume w.l.o.g. that $N_{\theta,\lambda}(i,j) \cap S \subset R$ and $N_{\theta,\lambda}(i+10,j) \cap S \subset B$ (the other case is symmetric); see Figure 1. In this case, we add edges to G between r^* and every point in $T_{\theta,\lambda}(i+10,j) \cap B$, between b^* and every point in $T_{\theta,\lambda}(i,j) \cap R$, as well as the edge r^*b^*. In particular, the edges added to G form a *bistar* centered at r^* and b^*.

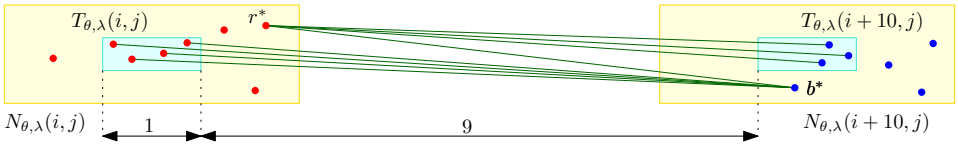


Figure 1 Case 1: Both $N_{\theta,\lambda}(i,j)$ and $N_{\theta,\lambda}(i+10,j)$ are monochromatic.

- **Case 2:** Exactly one of $N_{\theta,\lambda}(i,j)$ and $N_{\theta,\lambda}(i+10,j)$ is monochromatic. Assume w.l.o.g. that $N_{\theta,\lambda}(i,j)$ is monochromatic and $N_{\theta,\lambda}(i+10,j)$ is bichromatic; see Figure 2 (the other case is symmetric). If $N_{\theta,\lambda}(i,j) \cap S \subset R$, we add an edge to G between b^* and every point in $T_{\theta,\lambda}(i,j) \cap R$. Otherwise, (when $N_{\theta,\lambda}(i,j) \cap S \subset B$), we add an edge to G between r^{**} and every point in $T_{\theta,\lambda}(i,j) \cap B$. In this case, the edges added to G form a *star*.

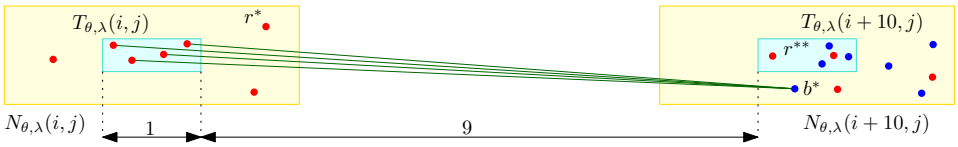


Figure 2 Case 2: $N_{\theta,\lambda}(i,j)$ is monochromatic and $N_{\theta,\lambda}(i+10,j)$ is bichromatic.

- **Case 3:** Both $N_{\theta,\lambda}(i,j)$ and $N_{\theta,\lambda}(i+10,j)$ are bichromatic; see Figure 3. In this case, we add two edges to G: the edges r^*b^* and $r^{**}b^{**}$.

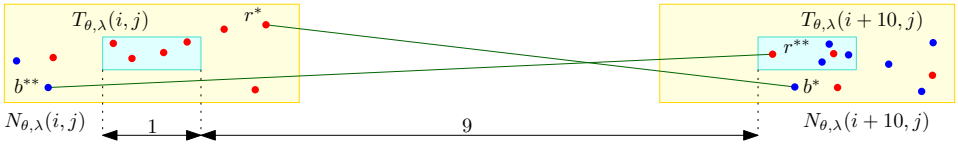


Figure 3 Case 3: Both $N_{\theta,\lambda}(i,j)$ and $N_{\theta,\lambda}(i+10,j)$ are bichromatic.

This completes the construction of the bipartite graph $G = (R \cup B, E)$. We show that G is a $(3+\varepsilon)$-spanner of $K(R,B)$ in Section 2.3, and that it contains $O(n/\sqrt{\varepsilon})$ edges in Section 2.4. We describe how to compute G in $O(\sqrt{1/\varepsilon} \cdot n \log n)$ time in Section 2.5.

2.3 Stretch Analysis

In this section, we prove that graph G is a $(3+\varepsilon)$-spanner of $K(R,B)$. We begin with a preliminary observation. The Euclidean length of an edge is given by the Pythagorean theorem for its x- and y-extents. However, if the y-extent is much smaller than the x-extent,

we can bound the Euclidean length from above by a polynomial as follows: For all $x \geq 1$ and $y > 0$, we have

$$\sqrt{x^2 + y^2} \leq x + y^2/2. \tag{3}$$

We can now prove the main result of this subsection.

▶ **Lemma 4.** *For every finite bichromatic set $S = R \cup B$ in the plane, the graph $G \subseteq K(R, B)$ constructed in Section 2.2 is $(3 + \varepsilon)$-spanner for $K(R, B)$.*

Proof. We proceed by induction on the length of the bichromatic edges in $K(R, B)$. Specifically, let $D = \{|rb| : (r, b) \in R \times B\}$ be the set of distances between red-blue pairs in S; and let $0 = d_0 < d_1 < \ldots < d_m$ be elements of $\{0\} \cup D$ in increasing order.

We prove, by induction on $\ell = 0, 1, \ldots, m$, that for all $(r, b) \in R \times B$ with $|rb| \leq d_\ell$, the graph G contains an rb-path of length at most $(3 + \varepsilon)|rb|$. In the base case, where $\ell = 0$ and $d_\ell = 0$, the claim vacuously holds, as there is no red-blue pair (r, b) with $|rb| = 0$.

In the induction step, let $d_\ell \in D$ and assume that for all $(r', b') \in R \times B$, if $|r'b'| < d_\ell$, then G contains an $r'b'$-path of length at most $(3+\varepsilon)|r'b'|$. Let $(r, b) \in R \times B$ with $|rb| = d_\ell$. By Lemma 3, there exist $\theta \in \Phi$, $\lambda \in \Lambda$ and $i, j \in \mathbb{Z}$ such that $T_{\theta,\lambda}(i, j)$ and $T_{\theta,\lambda}(i + 10, j)$ each contain one of r and b; or $T'_{\theta,\lambda}(i, j)$ and $T'_{\theta,\lambda}(i + 10, j)$ each contain one of r and b. To simplify our argument, we may assume w.l.o.g. that $\theta = 0$ and $\lambda = 1$, $r \in T_{\theta,\lambda}(i, j)$ and $b \in T_{\theta,\lambda}(i + 10, j)$. Under these assumptions, $N_{\theta,\lambda}(i, j) = N(i, j)$, which is an axis-aligned rectangle of width 3 and height 3δ. We distinguish between three cases:

- **Case 1:** Both $N(i, j)$ and $N(i + 10, j)$ are monochromatic; see Figure 1. By construction (Case 1), G contains the rb-path $P = (r, b^*, r^*, b)$. Since the pair r^* is the rightmost red point in $N(i, j)$ and b^* is the leftmost blue point in $N(i + 10, j)$, then we have $x(r) \leq x(r^*) < x(b^*) \leq x(b)$, where $x(r)$, $x(r^*)$, $x(b^*)$ and $x(b)$, resp., denote the x-coordinate of r, r^*, b^* and b. Then, the length of b^*r^* can be bounded above by $|b^*r^*| \leq \sqrt{(x(b^*) - x(r^*))^2 + (3\delta)^2} \leq \sqrt{(x(b) - x(r))^2 + (3\delta)^2}$. Using Equation (3) and $9 \leq |rb|$, it follows that

$$\begin{aligned} |b^*r^*| &\leq \sqrt{(x(b) - x(r))^2 + (3\delta)^2} \\ &\leq x(b) - x(r) + 9\delta^2/2 \\ &= x(b) - x(r) + 3\varepsilon/2 \\ &\leq (1 + \varepsilon/3)|rb|. \end{aligned}$$

  By a similar process, we obtain $|rb^*| \leq (1 + \varepsilon/3)|rb|$ and $|r^*b| \leq (1 + \varepsilon/3)|rb|$. Thus, the length of the rb-path P is at most $(3 + \varepsilon)|rb|$ since $|P| = |rb^*| + |b^*r^*| + |r^*b| \leq 3(1+\varepsilon/3)|rb| = (3+\varepsilon)|rb|$. (Note that we did not use the induction hypothesis in Case 1.)
- **Case 2:** One of $N(i, j)$ and $N(i + 10, j)$ is monochromatic and the other is bichromatic. Assume w.l.o.g. that $N(i, j)$ is monochromatic and $N(i+10, j)$ is bichromatic; see Figure 4 (the other case is symmetric). Let $\tilde{r} \in R \cap N(i + 10, j)$. Since $b^*, \tilde{r} \in N(i + 10, j)$, we have $|b^*\tilde{r}| \leq \operatorname{diam}(N(i + 10, j)) = 3 \operatorname{diam}(T(i, j))$. Furthermore, since $b \in T(i + 10, j)$, then the edge $\tilde{r}b$ lies in an axis-aligned rectangle spanned by $T(i + 10, j)$ and an adjacent tile, which implies $|\tilde{r}b| \leq 2 \operatorname{diam}(T(i, j))$. Note that $\operatorname{diam}(T(i, j)) = \sqrt{1 + \delta^2}$ by the Pythagorean Theorem. Equation (3) yields $\operatorname{diam}(T(i, j)) = \sqrt{1 + \delta^2} \leq 1 + \varepsilon/6$. Since $9 \leq |rb|$, it now follows that $|b^*\tilde{r}| \leq 3 \operatorname{diam}(T(i, j)) \leq 3(1 + \varepsilon/6) < 9 \leq |rb|$. Similarly, we have $|\tilde{r}b| < |rb|$.

  By the induction hypothesis, G contains some paths

$$P_1 = (b^*, \ldots, \tilde{r}) \quad \text{and} \quad P_2 = (\tilde{r}, \ldots, b),$$

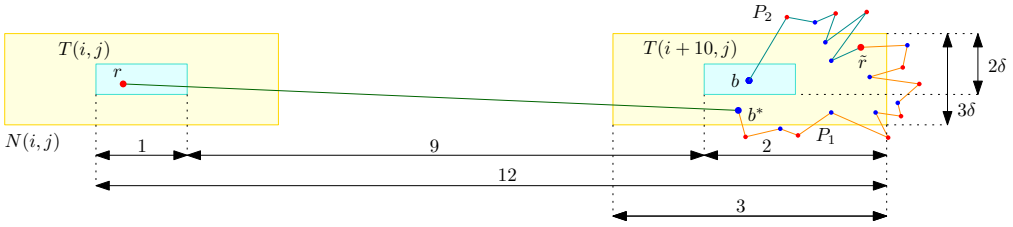


Figure 4 An rb-path when $N(i,j)$ is monochromatic and $N(i+10,j)$ is bichromatic.

resp., of length at most $(3+\varepsilon)|b^*\tilde{r}|$ and $(3+\varepsilon)|\tilde{r}b|$. Then, the concatenation of $rb^* \oplus P_1 \oplus P_2$ is an rb-path in G and its length can be bounded above as follows

$$\begin{aligned}
|rb^*| + |P_1| + |P_2| &\leq \sqrt{11^2 + (2\delta)^2} + (3+\varepsilon)|b^*\tilde{r}| + (3+\varepsilon)|\tilde{r}b| \\
&\leq 11 + 2\delta^2 + 3(3+\varepsilon)\operatorname{diam}(T(i,j)) + 2(3+\varepsilon)\operatorname{diam}(T(i,j)) \\
&= 11 + 2\varepsilon/3 + 5(3+\varepsilon)(1+\varepsilon/6) \\
&= 11 + 2\varepsilon/3 + 5(3 + 3\varepsilon/2 + \varepsilon^2/6).
\end{aligned}$$

Recall that $0 < \varepsilon < 1$, hence $\varepsilon^2 < \varepsilon$. It follows that

$$\begin{aligned}
|rb^*| + |P_1| + |P_2| &\leq 11 + 2\varepsilon/3 + 5(3 + 5\varepsilon/3) \\
&= 26 + 9\varepsilon \\
&< 9(3+\varepsilon) \\
&\leq (3+\varepsilon)|rb|.
\end{aligned}$$

So, the concatenation of $rb^* \oplus P_1 \oplus P_2$ is an rb-path in G of length at most $(3+\varepsilon)|rb|$.

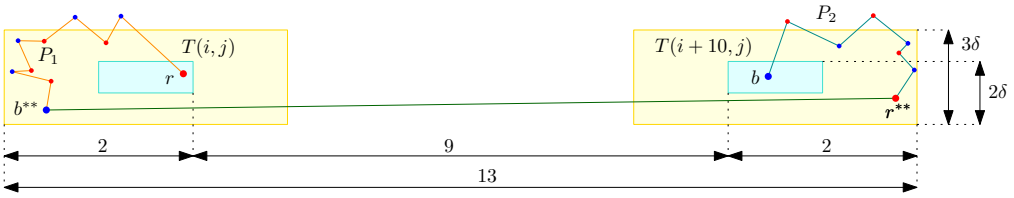


Figure 5 An rb-path when both $N(i,j)$ and $N(i+10,j)$ are bichromatic.

Case 3: Both $N(i,j)$ and $N(i+10,j)$ are bichromatic; see Figure 5. Similarly to Case 2, we have $|rb^{**}| \leq \operatorname{diam} N(i,j) < |rb|$ and $|r^{**}b| \leq \operatorname{diam} N(i+10,j) < |rb|$. By the induction hypothesis, G contains some paths

$$P_1 = (r, \ldots, b^{**}) \quad \text{and} \quad P_2 = (r^{**}, \ldots, b),$$

resp., of length at most $(3+\varepsilon)|rb^{**}|$ and $(3+\varepsilon)|r^{**}b|$. Then, the concatenation $P = P_1 \oplus b^{**}r^{**} \oplus P_2$ is an rb-path of G, and observe that by a similar process as in Case 2, its length be bounded above as follows

$$\begin{aligned}
|P| &= |P_1| + |b^{**}r^{**}| + |P_2| \\
&\leq (3+\varepsilon)|rb^{**}| + \operatorname{diam}(N(i,j) \cup N(i+10,j)) + (3+\varepsilon)|r^{**}b| \\
&\leq 2(3+\varepsilon)\operatorname{diam}(T(i,j)) + \sqrt{13^2 + (3\delta)^2} + 2(3+\varepsilon)\operatorname{diam}(T(i+10,j)) \\
&\leq 4(3+\varepsilon)(1+\varepsilon/6) + 13 + 3\varepsilon/2 \\
&\leq 4(3+5\varepsilon/3) + 13 + 3\varepsilon \\
&= 25 + 49\varepsilon/6 \\
&< 9(3+\varepsilon) \\
&\leq (3+\varepsilon)|rb|.
\end{aligned}$$

So, the concatenation $P_1 \oplus b^{**}r^{**} \oplus P_2$ is an rb-path in G of length at most $(3+\varepsilon)|rb|$. Overall, the bipartite graph G is a $(3+\varepsilon)$-spanner of $K(R,B)$. ◀

2.4 Sparsity Analysis

Fix an angle $\theta \in \Phi$. We distinguish between two types of edges in G. An edge is of *Type 1* if it was added when the construction processed a pair of tiles $T_{\theta,\lambda}(i,j), T_{\theta,\lambda}(i+10,j)$ where one of the tiles had monochromatic neighborhood; otherwise it is of *Type 2*. In particular, edges of Type 1 correspond to Cases 1-2 of the construction, and Type 2 to Case 3.

Overview. We count edges of each type separately in Lemmas 6 and 8 below. Edges of Type 1 are the main concern, since our spanner may contain many edges of Type 1 for each pair $T_{\theta,\lambda}(i,j), T_{\theta,\lambda}(i+10,j)$. However, even if the neighborhood of one of the tiles is monochromatic, their union $N_{\theta,\lambda}(i,j) \cup N_{\theta,\lambda}(i+10,j)$ contains both colors. We show (Lemma 5) that if we go up 26 scales higher, then every point in $T_{\theta,\lambda}(i,j) \cup T_{\theta,\lambda}(i+10,j)$ is contained in a bichromatic neighborhood: It follows that each point $s \in S$ is incident to an edge of Type 1 in $O(1)$ scales; hence there are $O(n)$ edges of Type 1 (for each $\theta \in \Phi$).

For every pair $T_{\theta,\lambda}(i,j), T_{\theta,\lambda}(i+10,j)$ with bichromatic neighborhoods, we add only two edges of Type 2, so it is enough to analyze the number of such pairs. We rely on the classic result that a quadtree-based WSPD has $m = O(n)$ well-separated pairs [20, Theorem 4.1.4]. The proof of this result hinges on properties of quadtrees and compressed quadtrees: The compressed quadtree for n points is known to have $O(n)$ nodes, but chains of single-child nodes in the uncompressed quadtree must be handled carefully. We adapt this approach to our pairs of tiles (Lemma 7).

It is easy to arrange the tiles $T_{\theta,\lambda}$, $\lambda \in \Lambda$, into groups in which the scales differ by integer factors – so we can obtain a laminar system, which supports a quadtree. However, for tiles in a standard quadtree, the translated tiles $T'_{\theta,\lambda}$ would not be laminar. For this reason, we use a variation of quadtrees: nonatrees [10], in which each tile is subdivided into nine congruent subtiles. Using nonatrees, the tiles in $T_{\theta,\lambda}$ and $T'_{\theta,\lambda}$, over all $\lambda \in \Lambda$, can be grouped into laminar systems.

Edges of Type 1. We introduce a charging scheme: We charge each edge st of Type 1 to one of its vertices as follows. Assume st was added to G when processing the pair tiles $T_{\theta,\lambda}(i,j)$ and $T_{\theta,\lambda}(i+10,j)$. If $N_{\theta,\lambda}(i,j)$ is monochromatic, we charge st to s, otherwise (when $N_{\theta,\lambda}(i+10,j)$ is monochromatic), we charge st to t.

▶ **Lemma 5.** *Let $s \in S$. Assume that s lies in tiles $T_{\theta,\lambda}(i,j)$ and $T_{\theta,\lambda'}(i',j')$ with monochromatic neighborhoods, where $\lambda = \gamma^p, \lambda' = \gamma^{p'}$ with $p < p'$, and our construction added an edge st of Type 1 when processing the pair of tiles $T_{\theta,\lambda}(i,j)$ and $T_{\theta,\lambda}(i+10,j)$. Then we have $p' < p + 26$.*

Proof. Assume w.l.o.g. that $\theta = 0$ and s is red (i.e., $s \in R$). Since we added an edge st for some $t \in N_{\theta,\lambda}(i+10,j)$, then t is blue. Note that

$$|x(s) - x(t)| \le |(i-1)\gamma^p - (i+12)\gamma^p| = 13\gamma^p < 3^{26/11}\,\gamma^p = \gamma^{p+26},$$
$$|y(s) - y(t)| \le \delta\gamma^p.$$

Recall that $s \in T_{\theta,\lambda'}(i,' j')$. For any $q \in \mathbb{R}^2$ such that $|x(s) - x(q)| \le \gamma^{p'}$ and $|y(s) - y(q)| \le \delta \cdot \gamma^{p'}$, we have $q \in N_{\theta,\lambda'}(i,' j')$. Hence $p' \ge p + 26$ implies $t \in N_{\theta,p'}(i,' j')$, and so the neighborhood $N_{\theta,d'}(i,' j')$ is bichromatic, and st cannot be charged to s. We conclude that $p' < p + 26$, as claimed. ◀

▶ **Lemma 6.** *For every bichromatic set $S = R \cup B$ of n points in the plane, and for every $\theta \in \Phi$, the graph G has $O(n)$ edges of Type 1.*

Proof. Each vertex $s \in S$ receives charges from at most 26 scales by Lemma 5. Consequently, there are $O(|S|) = O(n)$ edges of Type 1. ◀

Edges of Type 2. For the purpose of analyzing the edges of Type 2, we arrange the nonempty tiles $T_{\theta,\lambda}(i,j)$ into eleven nonatrees as follows. For $\sigma = 0, 1, \ldots, 10$, let

$$\Lambda_\sigma = \{\gamma^\sigma \cdot 3^p : p \in \mathbb{Z}\} = \left\{3^{\sigma/11} \cdot 3^p : p \in \mathbb{Z}\right\}.$$

Then the set of tiles $\{T_{\theta,\lambda}(i,j) : \lambda \in \Lambda_\sigma\}$ is a laminar set system (i.e., any two tiles are either interior-disjoint or one contains the other). Importantly, the translated tiles $\{T'_{\theta,\lambda}(i,j) : \lambda \in \Lambda_\sigma\}$ also form a laminar system [10] (this is why we use nonatrees, instead of quadtrees). Let $\mathcal{Q}_\sigma$ denote the set of tiles $T_{\theta,\lambda}(i,j)$ such that $\lambda \in \Lambda_\sigma$ and $T_{\theta,\lambda}(i,j) \cap S \neq \emptyset$ (the case of the tiles $T'_{\theta,\lambda}(i,j)$ is analoguous). Containment among the tiles in $\mathcal{Q}_\sigma$ defines a hierarchy (i.e., poset or tree); and we refer to $\mathcal{Q}_\sigma$ as a *nonatree*. We can now define a parent-child relationship between the tiles in $\mathcal{T}_\sigma$: For $T_{\theta,\lambda}(i,j), T_{\theta,\lambda'}(i',j') \in \mathcal{T}_\sigma$, we say that $T_{\theta,\lambda}(i,j)$ is a *child* of $T_{\theta,\lambda'}(i',j')$ if $T_{\theta,\lambda}(i,j) \subset T_{\theta,\lambda'}(i',j')$ and $\lambda' = 3\lambda$. It is clear from the definition that every tile has at most nine children. Let $\mathcal{R}_\sigma$ be the set of tiles $T \in \mathcal{Q}_\sigma$ that have two or more children. Containment among the tiles in $\mathcal{R}_\sigma$ gives a partition tree on S for every $\sigma \in \{0, \ldots, 10\}$, consequently

$$|\mathcal{R}_\sigma| \le n. \tag{4}$$

Note that every tile $T \in \mathcal{Q}_\sigma$ is contained in some tile $\widehat{T} \in \mathcal{R}_\sigma$; possibly $\widehat{T} = T$. For every $T \in \mathcal{Q}_\sigma$, let $\overline{p}(T)$ be the smallest tile in $\mathcal{R}_\sigma$ that contains T (i.e., $\overline{p}(T)$ is the closest ancestor of T in $\mathcal{R}_\sigma$).

We can now introduce a charging scheme: Consider the edges r^*b^* and $r^{**}b^{**}$ of Type 2 that were added to G when we processed the pair of tiles $\{T_1, T_2\} = \{T_{\theta,\lambda}(i,j), T_{\theta,\lambda}(i+10,j)\}$ with bichromatic neighborhoods. Let λ_1 and λ_2 be the scales of $\overline{p}(T_1)$ and $\overline{p}(T_2)$, respectively. If $\lambda_1 \le \lambda_2$, then we *charge* both edges to $\overline{p}(T_1)$; otherwise (if $\lambda_1 > \lambda_2$), we charge them to $\overline{p}(T_2)$.

▶ **Lemma 7.** *In the charging scheme above, every tile $T \in \bigcup_{\sigma=0}^{10} \mathcal{R}_\sigma$ receives $O(1)$ charges.*

Proof. Let $\sigma \in \{0, \ldots, 10\}$ and $T \in \mathcal{R}_\sigma$. We claim that T has received at most 148 units of charges. Suppose, for contradiction, that T receives 149 or more charges. Assume w.l.o.g. that $T = T_{\theta,\lambda}(i, j)$ with $\theta = \lambda = \sigma = 0$. In particular, T is an axis-aligned rectangle of $\text{width}(T) = 1$.

Let $\mathcal{P}_1$ be the collection of pairs of tiles (T_1, T_2) such that two bichromatic edges corresponding to (T_1, T_2) are charged to T and $\overline{p}(T_1) = T$. By assumption, we have $|\mathcal{P}_1| \geq \lceil 149/2 \rceil = 75$. In each pair $(T_1, T_2) \in \mathcal{P}_1$, the tile T_1 is either *left* (i.e., $T_1 = T(i, j)$ and $T_2 = T(i + 10, j)$) or *right* (i.e., $T_1 = T(i + 10, j)$ and $T_2 = T(i, j)$). We may assume w.l.o.g. that in at least half of these pairs, T_1 is left. Let $\mathcal{P}_2 \subseteq \mathcal{P}_1$ be the set of these pairs; note that $|\mathcal{P}_2| \geq \lceil |\mathcal{P}_1|/2 \rceil = 38$.

For each pair $(T_1, T_2) \in \mathcal{P}_2$, either $T_1 = T$ of the nonatree $\mathcal{Q}_\sigma$ contains a descending chain $\text{chain}(T_1)$ from T to T_1 where all intermediate tiles have only one child (i.e., they are in $\mathcal{Q}_\sigma \setminus \mathcal{R}_\sigma$). There is at most one pair $(T_1, T_2) \in \mathcal{P}_2$ such that $T_1 = T$; so T_1 is a strict descendant of T in at least 37 pairs in $\mathcal{P}_2$. If $T_1 \neq T$, then $\text{chain}(T_1)$ includes exactly one child of T. Since T has at most nine children in $\mathcal{Q}_\sigma$, there are at least $\lceil 37/9 \rceil = 5$ pairs in $\mathcal{P}_2$ where $\text{chain}(T_1)$ includes the same child of T; let T_{child} denote this child of T and let $\mathcal{P}_3$ be the set of these pairs; see Figure 6.

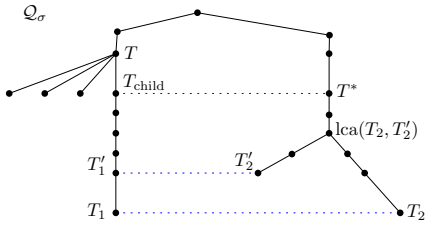


Figure 6 A schematic layout of the nodes in the nonatree $\mathcal{Q}_\sigma$.

Note for all pairs $(T_1, T_2) \in \mathcal{P}_3$, the tiles T_1 are at different scales (since they are in the same single-child chain). Let (T_1, T_2) and (T_1', T_2') be the pairs in $\mathcal{P}_3$ at the lowest and the second lowest scales, respectively. Since $|\mathcal{P}_3| \geq 5$ and $\text{width}(T) = 1$, then $\text{width}(T_1) \leq 3^{-5}$ and $\text{width}(T_1') \leq 3^{-4}$. Since $T_1 \subseteq T_1'$, then $\text{dist}(T_1', T_2) \leq \text{dist}(T_1, T_2) = 9\,\text{width}(T_1) \leq 3\,\text{width}(T_1')$. It follows that

$$\begin{aligned}
\text{dist}(T_2, T_2') &\geq \text{dist}(T_1', T_2') - \text{dist}(T_1', T_2) - \text{width}(T_2) \\
&\geq \text{dist}(T_1', T_2') - \text{dist}(T_1, T_2) - \text{width}(T_2) \\
&\geq \left(9 - 3 - \frac{1}{3}\right) \text{width}(T_1') > 5\,\text{width}(T_1') > 0,
\end{aligned}$$

and, in particular, the tiles T_2 and T_2' are disjoint. Furthermore, $T_1 \subset T_1' \subseteq T_{\text{child}}$ implies

$$\begin{aligned}
\text{dist}(T_{\text{child}}, T_2) &\leq \text{dist}(T_1, T_2) = 9\,\text{width}(T_1) \leq \frac{9}{3^5} < \frac{1}{9}, \\
\text{dist}(T_{\text{child}}, T_2') &\leq \text{dist}(T_1', T_2') = 9\,\text{width}(T_1') \leq \frac{9}{3^4} < \frac{1}{3}.
\end{aligned}$$

As $\text{width}(T_{\text{child}}) = \frac{1}{3}$, both T_2 and T_2' are contained in some tile $T^* \in \mathcal{Q}_\sigma$ at the same scale as T_{child}. But $\text{chain}(T_1)$ is a single-child chain of tiles, so $T_2, T_2' \not\subseteq T_{\text{child}}$, and $T^* \neq T_{\text{child}}$ (although, T^* could be a sibling of T_{child}). Note that the least common ancestor $\text{lca}(T_2, T_2')$ of T_2 and T_2' in the nonatree $\mathcal{Q}_\sigma$ satisfies $\text{lca}(T_2, T_2') \subseteq T^*$. In particular, $\text{lca}(T_2, T_2')$ is at a scale below T; and $\text{lca}(T_2, T_2')$ has at least two children, which implies $\text{lca}(T_2, T_2') \in \mathcal{R}_\sigma$.

Consequently, both $\overline{p}(T_2)$ and $\overline{p}(T_2')$ are at a scale below $T = \overline{p}(T_1) = \overline{p}(T_1')$; and so our charging scheme could not have charged (T_1, T_2) and (T_1', T_2') to T: A contradiction. We conclude that at most 148 edges were charged to the tile T. ◀

▶ **Lemma 8.** *For every bichromatic set $S = R \cup B$ of n points in the plane, and for every $\theta \in \Phi$, the graph G has $O(n)$ edges of Type 2.*

Proof. This readily follows from the combination of Equation (4) and Lemma 7. ◀

2.5 Running Time Analysis

For every set S of n bichromatic points in the plane and every $\varepsilon > 0$, our spanner construction (Section 2.2) can be implemented in $O(\sqrt{1/\varepsilon} \cdot n \log n)$ time. The construction is done independently for each $\theta \in \Phi$, where $|\Phi| = O(\sqrt{1/\varepsilon})$, so it is enough to show that the construction takes $O(n \log n)$ time for each $\theta \in \Phi$. Assume w.l.o.g. that $\theta = 0$. Instead of infinite tilings, we can use nonatrees (the generalization of quadtrees, where each cell is split into 9 congruent cells) after an affine transformation $f(x, y) = (x, y/\delta)$, which maps an $1 \times \delta$ rectangle into a unit square. Let $S_0 := f(S)$. Let B_0 be an axis-aligned bounding box of S_0 where $\text{width}(B_0) = 2\,\text{height}(B_0)$, and scale B_0 centrally up by factors of $3^{\sigma/11}$ for $\sigma = 0, \ldots, 10$. Each scaled copy of B_0 can be expressed as an intersection of two axis-aligned squares $B_0 = B \cap B'$, where $\text{width}(B) = \text{width}(B') = \text{width}(B_0)$; note that $B' = B + (0, \frac{1}{2}\,\text{height}(B))$.

Starting from each of these 22 squares (B and B' with $\sigma = 0, \ldots, 10$), as root cells, we can build compressed nonatrees $\mathcal{R}_\sigma$ and $\mathcal{R}_\sigma'$ for S_0 in $O(n \log n)$ time; analogously to the construction of a quadtrees [10, 9, 20]. The compressed nonatree for n points has $O(n)$ cells, but the (uncompressed) nonatree can have arbitrarily many cells (as it may have descending chains of arbitrarily length). By Lemmas 5 and 7, our construction only uses (uncompressed) nonatree nodes that have an ascending chain of length at most 5 to a nodes of the compressed nonatree $\mathcal{R}_\sigma$: We augment $\mathcal{R}_\sigma$ with up to 5 single-child descendants per node to obtain a tree $\mathcal{R}_\sigma^*$ with $O(n)$ nodes. In a bottom-up traversal of $\mathcal{R}_\sigma^*$, we can find the leftmost and rightmost red (blue) points in each cell in $O(n)$ time. For each cell $T(i, j)$, we can find their eight adjacent cells in $N(i, j)$ in $O(\log n)$ time, and classify $N(i, j)$ as either monochromatic or bichromatic. In another bottom-up traversal, we find all pairs of tiles $T(i, j)$ and $T(i + 10, j)$ and find the edges of G, according to Cases 1–3 in Section 2.2, in $O(n)$ time.

2.6 Generalization to Higher Dimensions

It is straightforward to generalize our construction to Euclidean d-space: In $\mathbb{R}^2$, we used basic tiles of size $1 \times \sqrt{\varepsilon/3}$, and rotated the tilings with a set $\Phi \subset [0, \pi]$ of $\Theta(\sqrt{1/\varepsilon})$ angles at distance $\arctan(\sqrt{\varepsilon/3}/22) = O(\sqrt{\varepsilon})$ apart. In d-space, for $d \geq 3$, we use basic tiles of size $1 \times \sqrt{\varepsilon/(d+1)} \times \ldots \times \sqrt{\varepsilon/(d+1)}$, and we use a set Φ of rotations constructed as follows: We choose a set $U_d \subset \mathbb{S}^{d-1}$ of unit vectors such that the $O(\sqrt{\varepsilon})$-neighborhoods of the vectors in $\mathbb{S}^{d-1}$ (where each neighborhood is a spherical cap) cover U_d. A standard packing argument shows that $O(1/\varepsilon^{(d-1)/2})$ directions suffice. Then for each $\vec{u} \in U_d$, we rotate the basic tiling such such that the positive x_1-axis is parallel to $\vec{u}$. With this setup, Lemma 3, generalizes. The construction, as well as the stretch and sparsity analyses readily extend to $\mathbb{R}^d$. The number of edges is $O(|U_d| \cdot n) = O(n/\varepsilon^{(d-1)/2})$.

3 Bichromatic Spanners on the Line

In this section, we study spanners for n bichromatic points on the real line $\mathbb{R}$. The lower bound construction in [15] shows, that for every $\varepsilon > 0$, there is a bichromatic point set such that a $(3-\varepsilon)$-spanner must contain all edges of the complete bipartite graph $K(A, B)$; this construction consists of two monochromatic clusters of diameter $\frac{\varepsilon}{3}$ at unit distance apart, and can be realized already in 1-dimension. We show that there is 3-spanner with at most $2n-3$ edges (Proposition 9; furthermore, the minimum spanning tree of $K(R, B)$ (denoted minBST(S)) is a 7-spanner, and stretch bound 7 is the best possible (Proposition 10).

We start by introducing some notation. Let $S = R \cup B$ be a set of n bichromatic points on a line. Assume that $S = \{s_1, \ldots, s_n\}$, where the points are labeled in linear order along the line. Partition S into maximal subsets of consecutive points of the same color $S = \bigcup_{i=1}^{k} S_i$; we call these subsets *blocks*. Clearly, the blocks span disjoint intervals along the line, which are also ordered linearly. For any nonempty subset $S' \subseteq S$, denote by $\ell(S')$ and $r(S')$ the leftmost and rightmost point in S', respectively.

▶ **Proposition 9.** *For a set $S = B \cup R$ of n bichromatic points on a line, there is a 3-spanner with at most $2n-3$ edges that can be constructed in $O(n \log n)$ time.*

Proof. We construct a subgraph H of $K(A, B)$ as follows. Consider the partition of S into blocks, $S = \bigcup_{i=1}^{k} S_i$. For every $i = 1, \ldots, k$, and for every point $s \in S_i$, add edges $sr(S_{i-1})$ and $s\ell(S_{i+1})$ to H (if these edges exist). In particular, we add only one edge for every $s \in S_1 \cup S_k$, and two edges for all other vertices. Note also that the $k-1$ edges of the form $r(S_i)\ell(S_{i+1})$ are counted twice. Since $|S_1| \geq 1$, $|S_k| \geq 1$, and $k \geq 2$, then the total number of edges in H is $2n - (|S_1| + |S_k|) - (k-1) \leq 2n - 2 - 1 = 2n - 3$.

Next we show that H is a 3-spanner. Let $a, b \in S$ be two points of opposite colors, and assume w.l.o.g. that $a \in S_i$, $b \in S_j$ for some $1 \leq i < j \leq k$. Then H contains the path

$$P = (a, \ell(S_{i+1}), r(S_i), r(S_{i+1}), \ldots, r(S_{j-1}), b).$$

The first three edges of the path P intersect in the interval, $r(S_i), \ell(S_{i+1})$. All subsequent edges are pairwise interior disjoint (from each other and from the first three edges). This implies that every point on the line is contained in at most 3 edges of P, which immediately implies that $|P| \leq 3\,|ab|$, as required. ◀

Bichromatic Minimum Spanning Tree. Bandyapadhyay et al. [7] showed that if n bichromatic points are already sorted along the line, then minBST(S) can be computed in $O(n)$ time. (Biniaz et al. [11] showed how to compute minBST(S) in $O(n \log n)$ in the plane.)

We start with a characterization of minBST(S), based on the observations in [7]. Consider a block S_i, $1 \leq i \leq k$, defined above. For every point $s \in S_i$, the closest point of the opposite color is either $r(S_{i-1})$ or $\ell(S_{i+1})$. We define a partition $S_i = S_i^- \cup S_i^+$, where S_i^- (resp., S_i^+) is the set of points in S_i for which the closest point of the opposite color is in S_{i-1} (resp., S_{i+1}). Note that $S_1^- = \emptyset$ and $S_k^+ = \emptyset$, but S_i^- or S_i^+ may be empty for any $1 < i < k$.

Recall, that if $G = (V, E, w)$ is an edge-weighted connected graph with positive edge weights, then for every nontrivial cut $V = V_1 \cup V_2$, the MST of G contains a minimum-weight edge between V_1 and V_2. We apply this observation for nontrivial cuts $S = \{s_1, \ldots, s_j\} \cup \{s_{j+1}, \ldots, s_n\}$ for $i = 1, \ldots, n-1$ in the complete bipartite graph $K(R, B)$. Specifically, if s_j and s_{j+1} are in different blocks, then the shortest bichromatic edge across the cut is $s_j s_{j+1}$; in this case $s_j = r(S_i)$ and $\ell(S_{i+1})$ for some $1 \leq i \leq k$. Now assume that both s_j and s_{j+1} are in the same block S_i. If $s_j, s_{j+1} \in S_i^-$ (resp., $s_j, s_{j+1} \in S_i^+$), then $r(S_{i-1})s_j$

(resp., $\ell(S_{j+1})s_{i+1}$) is the shortest bichromatic edge across the cut. Finally, if $s_i \in S_j^-$ and $s_{i+1} \in S_j^+$, then the shortest bichromatic edge across the cut is the shorter of two candidates: $r(S_{i-1})s_j$ or $\ell(S_{j+1})s_{i+1}$. Note that we considered $n-1$ nontrivial cuts, and obtained $n-1$ distinct edges: Let H be the graph formed by these edges. Since H has $n-1$ edges and it is a subgraph of $\mathrm{minBST}(S)$, then $H = \mathrm{minBST}(S)$.

We give a tight bound on the stretch of the bichromatic MST on a line.

▶ **Proposition 10.** *For a set $S = B \cup R$ of n bichromatic points on a line, $\mathrm{minBST}(S)$ is a 7-spanner; and for every $\varepsilon > 0$, there is a bichromatic point set $S \subset \mathbb{R}$ such that the stretch of $\mathrm{minBST}(S)$ is at least $7 - \varepsilon$.*

Proof. It is enough to show that for any pair of points a, b in consecutive blocks, $\mathrm{minBST}(S)$ contains a path of length at most $7\,|ab|$. Indeed, if $a \in S_i$ and $b \in S_j$, where $i < j$, then we can first consider the path $P = (a, \ell(S_{i+1}), \ldots, \ell(S_{j-1}), b)$ in $K(R, B)$. Note that $|P| = |ab|$, and every edge of P is between points in consecutive blocks. By replacing every edge $e \subseteq P$ by a path in minBST of length at most $7\,|e|$, we obtain an ab-path in minBST of length at most $\sum_{e \subseteq P} 7\,|e| = 7\,|P| = 7\,|ab|$.

We may assume w.l.o.g. that $a \in S_i$ and $b \in S_{i+1}$. We claim that $\mathrm{minBST}(S)$ contains a path P_1 from a to $\ell(S_{i+1})$ of length $|P_1| \le 3\,|ab|$; and symmetrically a path P_2 from $r(S_i)$ to b of length $|P_2| \le 3\,|ab|$. Then the concatenation $P_1 \oplus \ell(S_{i+1})r(S_i) \oplus P_2$ is an ab-path in $\mathrm{minBST}(S)$, and its length is at most $|P_1| + |\ell(S_{i+1})r(S_i)| + |P_2| \le (3+1+3)|ab| = 7\,|ab|$, as required. It remains the prove the claim. We distinguish between two cases:

- **Case 1: $a \in S_i^+$ or $a = r(S_i^-)$ and $r(S_i^-)\ell(S_{i+1}) \in \mathrm{minBST}(S)$.** In this case, P_1 is a single edge path, and $|P_1| = |a\,\ell(S_{i+1})| \le |ab|$.
- **Case 2: $a \in S_i^-$ and if $a = r(S_i^-)$, then $r(S_i^-)\ell(S_{i+1}) \notin \mathrm{minBST}(S)$.** The latter condition implies that $r(S_{i-1})\ell(S_i^+) \in \mathrm{minBST}(S)$ because $r(S_{i-1})\ell(S_i^+)$ is shorter than $r(S_i^-)\ell(S_{i+1})$. In particular, we have $|r(S_{i-1})\ell(S_i^+)| \le |r(S_i^-)\ell(S_{i+1})| \le |ab|$. In this case, let $P_1 = (a, r(S_{i-1}), \ell(S_i^+), r(S_{i+1}))$. Now each of the three edges of P_1 has length at most $|ab|$, and so $|P_1| \le 3\,|ab|$, as required.

This confirms the claim in both cases; and completes the proof of the upper bound.

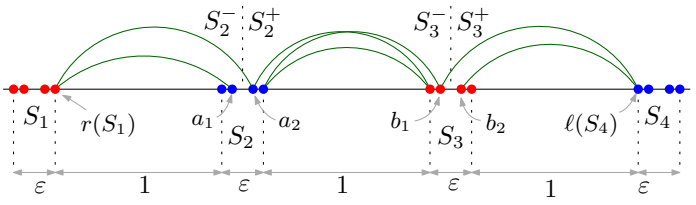


Figure 7 Lower bound construction.

For a lower bound consider a bichromatic point set with 4 blocks, $S = \bigcup_{i=1}^4 S_i$, each of 4 points, such that each block spans an interval of length $\varepsilon > 0$ for a sufficiently small $\varepsilon > 0$, and the distance between any two consecutive blocks is 1; see Figure 7. Let a_1, a_2 be the two interior points in S_2 such that $a_1 \in S_2^-$, $a_2 \in S_2^+$, and $|r(S_1)a_2| < |a_1\ell(S_3)|$. Symmetrically, let b_1, b_2 be the two interior points in S_3 such that $b_1 \in S_3^-$, $b_2 \in S_3^+$, and $|b_1\ell(S_4)| < |r(S_2)b_2|$. Then a_1 has degree 1 in $\mathrm{minBST}(S)$, and its only neighbor is $r(S_1)$. Similarly, the only neighbor of b_2 is $\ell(S_4)$. The set $S_2^+ \cup S_3^-$ induces a bistar centered at $r(S_2)$ and $\ell(S_3)$: A bistar on 4 vertices forms a spanning path on $S_2^+ \cup S_3^-$ between a_2 and b_1. Overall, the a_1b_2-path in $\mathrm{minBST}(S)$ is

$$P = (a_1, r(S_1), a_2, \ell(S_3), r(S_2), b_1, \ell(S_4), b_2).$$

We have $|a_1b_2| \le 1 + 2\varepsilon$, but every edge of P has length at least 1. Consequently the stretch of $\mathrm{minBST}(S)$ is at least $|P|/|a_1b_2| \ge 7/(1+2\varepsilon)$, which tends to 7 as $\varepsilon \to 0$. ◀

4 Conclusions

We conclude with a few open problems. What is the tight dependence on $\varepsilon > 0$ for the size of a $(3+\varepsilon)$-spanner for n bichromatic points in the plane? We have proved an upper bounds of $O(\sqrt{1/\varepsilon} \cdot n)$ in the plane, and $2n-3$ (i.e., no dependence on ε) on the line.

The *lightness* of a spanner H for an edge-weighted graph G is the ratio $w(H)/w(\mathrm{MST}(G))$ of the weight of the spanner to the weight of a minimum spanning tree of G. On bichromatic points on the line, we have shown (Proposition 10) that there is a 7-spanner of lightness 1. It remains an open to obtain upper bounds for stretch $3 \leq t < 7$ on the line, and for any $t > 3$ in the plane.

Our results give existential bounds on the size of a bichromatic spanners. The corresponding *optimization problems* remain open: For a given bichromatic point set $S = R \cup B$ and stretch $t \geq 1$, find a t-spanner of $K(R, S)$ of minimum size or minimum weight (note that for a given instance, stretch $1 \leq t < 3$ can be feasible). We conjecture that these problems are NP-hard already in Euclidean plane. Finding good approximations is left for future research. For monochromatic spanners in the plane, these optimization problems are known to be NP-hard, and bicriteria approximation algorithms were obtained only recently [24]. However, for bichromatic points, the optimization problem is already open for n red-blue points on the line with stretch $1 < t < 7$.

References

1. A. Karim Abu-Affash, Sujoy Bhore, Paz Carmi, and Dibyayan Chakraborty. Bottleneck bichromatic full Steiner trees. *Inf. Process. Lett.*, 142:14–19, 2019. doi:10.1016/J.IPL.2018.10.003.
2. A. Karim Abu-Affash, Sujoy Bhore, Paz Carmi, and Joseph S. B. Mitchell. Planar bichromatic bottleneck spanning trees. *J. Comput. Geom.*, 12(1):109–127, 2021. doi:10.20382/JOCG.V12I1A5.
3. Pankaj K. Agarwal, Herbert Edelsbrunner, and Otfried Schwarzkopf. Euclidean minimum spanning trees and bichromatic closest pairs. *Discret. Comput. Geom.*, 6:407–422, 1991. doi:10.1007/BF02574698.
4. Hugo A. Akitaya, Ahmad Biniaz, Erik D. Demaine, Linda Kleist, Frederick Stock, and Csaba D. Tóth. Minimum plane bichromatic spanning trees. *ACM Trans. Algorithms*, 21(4):48:1–48:14, 2025. doi:10.1145/3747591.
5. Sang Won Bae, Nicolau Oliver, and Evanthia Papadopoulou. Higher-order color Voronoi diagrams and the colorful Clarkson-Shor framework. In *Proc. 41st Symposium on Computational Geometry (SoCG)*, LIPIcs, pages 12:1–12:19. Schloss Dagstuhl, 2025. doi:10.4230/LIPIcs.SOCG.2025.12.
6. Andreas Baltz and Anand Srivastav. Approximation algorithms for the Euclidean bipartite TSP. *Oper. Res. Lett.*, 33(4):403–410, 2005. doi:10.1016/J.ORL.2004.08.002.
7. Sayan Bandyapadhyay, Aritra Banik, Sujoy Bhore, and Martin Nöllenburg. Geometric planar networks on bichromatic collinear points. *Theor. Comput. Sci.*, 895:124–136, 2021. doi:10.1016/J.TCS.2021.09.035.
8. Sergey Bereg, Yuya Higashikawa, Naoki Katoh, Junichi Teruyama, Yuki Tokuni, and Binhai Zhu. Constructing red-black spanners for mixed-charging vehicular networks. *Theor. Comput. Sci.*, 1023:114932, 2025. doi:10.1016/J.TCS.2024.114932.
9. Mark de Berg, Otfried Cheong, Marc J. van Kreveld, and Mark H. Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3rd edition, 2008. doi:10.1007/978-3-540-77974-2.
10. Sujoy Bhore, Martin Nöllenburg, Csaba D. Tóth, and Jules Wulms. Fully dynamic maximum independent sets of disks in polylogarithmic update time. *Discret. Comput. Geom.*, 75(2):391–430, 2026. doi:10.1007/S00454-025-00793-8.

11 Ahmad Biniaz, Prosenjit Bose, David Eppstein, Anil Maheshwari, Pat Morin, and Michiel H. M. Smid. Spanning trees in multipartite geometric graphs. *Algorithmica*, 80(11):3177–3191, 2018. doi:10.1007/S00453-017-0375-4.

12 Ahmad Biniaz, Prosenjit Bose, Anil Maheshwari, and Michiel H. M. Smid. Plane bichromatic trees of low degree. *Discret. Comput. Geom.*, 59(4):864–885, 2018. doi:10.1007/S00454-017-9881-Z.

13 Ahmad Biniaz, Anil Maheshwari, and Michiel H. M. Smid. Bottleneck bichromatic plane matching of points. In *Proc. 26th Canadian Conference on Computational Geometry (CCCG)*. Carleton University, Ottawa, ON, 2014. URL: http://www.cccg.ca/proceedings/2014/papers/paper63.pdf.

14 Magdalene G. Borgelt, Marc J. van Kreveld, Maarten Löffler, Jun Luo, Damian Merrick, Rodrigo I. Silveira, and Mostafa Vahedi. Planar bichromatic minimum spanning trees. *J. Discrete Algorithms*, 7(4):469–478, 2009. doi:10.1016/J.JDA.2008.08.001.

15 Prosenjit Bose, Paz Carmi, Mathieu Couture, Anil Maheshwari, Pat Morin, and Michiel H. M. Smid. Spanners of complete k-partite geometric graphs. *SIAM J. Comput.*, 38(5):1803–1820, 2009. doi:10.1137/070707130.

16 Prosenjit Bose, Paz Carmi, Mathieu Couture, Anil Maheshwari, Michiel H. M. Smid, and Norbert Zeh. Geometric spanners with small chromatic number. *Comput. Geom.*, 42(2):134–146, 2009. doi:10.1016/J.COMGEO.2008.04.003.

17 Paul B. Callahan and S. Rao Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *J. ACM*, 42(1):67–90, 1995. doi:10.1145/200836.200853.

18 Kenneth L. Clarkson. Approximation algorithms for shortest path motion planning. In *Proc. 19th ACM Symposium on Theory of Computing (STOC)*, pages 56–65, 1987. doi:10.1145/28395.28402.

19 John Fischer and Sariel Har-Peled. Dynamic well-separated pair decomposition made easy. In *Proc. 17th Canadian Conf. Comput. Geom. (CCCG)*, pages 235–238, Windsor, ON, 2005. URL: http://www.cccg.ca/proceedings/2005/32.pdf.

20 Sariel Har-Peled. *Geometric Approximation Algorithms*, volume 173 of *Mathematical Surveys and Monographs*. AMS, Boston, MA, 2011. doi:10.1090/surv/173.

21 Sariel Har-Peled and Manor Mendel. Fast construction of nets in low-dimensional metrics and their applications. *SIAM Journal on Computing*, 35(5):1148–1184, 2006. doi:10.1137/s0097539704446281.

22 Jeremy Karp and R. Ravi. A 9/7-approximation algorithm for graphic TSP in cubic bipartite graphs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, LIPIcs, pages 284–296. Schloss Dagstuhl, 2014. doi:10.4230/LIPIcs.APPROX-RANDOM.2014.284.

23 J. Mark Keil. Approximating the complete Euclidean graph. In *Proc. 1st Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 318 of *LNCS*, pages 208–213. Springer, 1988. doi:10.1007/3-540-19487-8_23.

24 Hung Le, Shay Solomon, Cuong Than, Csaba D. Tóth, and Tianyi Zhang. Approximate light spanners in planar graphs. In *Proc. 37th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2669–2701, 2026. doi:10.1137/1.9781611978971.98.

25 Ioannis Mantas, Evanthia Papadopoulou, Rodrigo I. Silveira, and Zeyu Wang. The farthest color Voronoi diagram in the plane. *Algorithmica*, 87(10):1393–1419, 2025. doi:10.1007/S00453-025-01311-1.

26 Giri Narasimhan and Michiel Smid. *Geometric Spanner Networks*. Cambridge University Press, 2007. doi:10.1017/cbo9780511546884.

27 Jesper Nederlof. Bipartite TSP in $o(1.9999^n)$ time, assuming quadratic time matrix multiplication. In *Proc. 52nd ACM Symposium on Theory of Computing (STOC)*, pages 40–53, 2020. doi:10.1145/3357713.3384264.

Improved Approximation Algorithms for Bounded-Length Path Augmentation

Felix Hommelsheim[1]
University of Cologne, Germany

Abstract

The Forest Augmentation Problem (FAP) asks for a minimum set of additional edges (links) that makes a given forest 2-edge-connected while spanning all vertices. A key special case is the Path Augmentation Problem (PAP), where the input forest consists of vertex-disjoint paths. Grandoni, Jabal Ameli, and Traub [STOC'22] recently broke the long-standing 2-approximation barrier for FAP, achieving a 1.9973-approximation. A crucial component of this result was their 1.9913-approximation for PAP; the first better-than-2 approximation for PAP.

In this work, we present a 1.9412-approximation for bounded-length PAP, in which each path of the input forest has constant length $p_{\max}$. The running-time of our algorithm is $O(n^{p_{\max}})$. One of our key innovations is a $(\frac{11}{6} + \varepsilon)$-approximation preserving reduction to so-called structured instances, which simplifies the problem and enables our improved approximation. Additionally, we introduce a new relaxation inspired by 2-edge covers and analyze it via a corresponding packing problem, where the relationship between the two problems is similar to the relationship between 2-edge covers and 2-matchings. Using a factor-revealing LP, we bound the cost of our solution to the packing problem w.r.t. the relaxation and derive a strong initial solution. We then transform this solution into a feasible PAP solution, combining techniques from FAP and related connectivity augmentation problems, along with new insights.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis; Theory of computation → Graph algorithms analysis; Theory of computation → Routing and network design problems

Keywords and phrases Approximation algorithms, path augmentation, connectivity augmentation, survivable network design

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.7

Category APPROX

Related Version *Full Version*: `https://arxiv.org/abs/2505.15324` [28]

Funding The work of the first author is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) by the grant HO 7562/2-1 (project ID: 573939419).

1 Introduction

Designing reliable networks that can withstand failures is a fundamental challenge in real-world applications. The field of *survivable network design* (SND) aims to construct cost-efficient networks that maintain connectivity even when some edges fail. Many natural SND problems are NP-hard (even APX-hard), justifying the study of approximation algorithms. A landmark result in this area is Jain's 2-approximation algorithm [31], which applies to a broad class of connectivity problems. However, breaking the 2-approximation barrier remains a major open question, even for many special cases of SND.

[1] A major part of this work was done while the author was affiliated with the University of Bremen, Germany.


© Felix Hommelsheim;
licensed under Creative Commons License CC-BY 4.0

In this paper, we focus on the Path Augmentation Problem (PAP), which is a special case of the Forest Augmentation Problem (FAP), a fundamental network augmentation problem. In the Forest Augmentation Problem we are given an undirected graph $G = (V, F \cup L)$, where F is a forest and $L \subseteq \binom{V}{2}$ is a set of additional edges (links) and the goal is to find a minimum-cardinality subset $S \subseteq L$ such that $(V, F \cup S)$ is 2-edge-connected[2]. The assumption that F is a forest is no restriction, as one can obtain an equivalent instance by contracting each 2-edge-connected component of F into a single vertex. The Path Augmentation Problem (PAP) is a special case of FAP where the given forest is composed of vertex-disjoint paths. We formally denote an instance of PAP by $G = (V, E(\mathcal{P}) \cup L)$, where $\mathcal{P}$ is the set of vertex-disjoint paths and $E(\mathcal{P})$ denotes its corresponding edge set. The goal is to find a minimum-cardinality subset $S \subseteq L$ such that $(V, E(\mathcal{P}) \cup S)$ is 2-edge-connected. Both problems are APX-hard [16, 18], ruling out the possibility of a Polynomial-Time Approximation Scheme (PTAS), unless P = NP. Until recently, no polynomial-time algorithm achieving a better-than-2 approximation was known for either problem.

Grandoni, Jabal Ameli, and Traub [25] were the first to break this barrier for PAP and FAP. In more detail, they obtained a 1.9913-approximation for PAP[3]. Combining this result with another result of [25] led to a 1.9973-approximation for FAP.

1.1 Our Contribution

In this paper, we consider *bounded-length PAP*, the special case of PAP in which each path in $\mathcal{P}$ contains at most $p_{\max}$ edges. This problem generalizes several well-established problems in connectivity augmentation, such as the unweighted 2-edge connected spanning subgraph problem (2ECSS), in which each path has length 0, or the Matching Augmentation Problem (MAP), in which each path has length 1; see Section 1.2 for more details. Our main result is a substantial improvement for bounded-length PAP.

▶ **Theorem 1.** *There is a* 1.9412*-approximation algorithm for PAP with running-time* $n^{O(p_{\max})}$.

To achieve this improvement, we introduce a new $(\frac{11}{6} + \varepsilon)$-approximation preserving reduction to structured instances, which is key to our approach. Additionally, we introduce a new relaxation of PAP, called 2-Edge Cover with Path Constraints (2ECPC), in which we have to compute a minimum-cardinality set of links $X \subseteq L$ such that $(V, E(\mathcal{P}) \cup X)$ is a 2-edge cover with the additional property that each path $P \in \mathcal{P}$ has two outgoing links. Instead of directly obtaining an approximate solution for 2ECPC, we consider a novel packing problem, called the Track Packing Problem (TPP). TPP is in some sense the corresponding packing problem to the covering problem 2ECPC, where the relationship between the two problems is inspired by the relationship between 2-edge covers and 2-matchings (precise definitions follow). We compute up to $|\mathcal{P}| + 2$ many solutions to TPP and analyze their cost using a factor-revealing linear program (LP). The resulting approximate solution serves as a suitable starting point for our framework. Later, this solution is turned into a feasible solution, highly relying on the fact that the input instance is structured. Our techniques also combine insights from [25] with methods used in other augmentation problems, together with new ideas tailored for structured instances of bounded-length PAP. We believe that

[2] A graph is k-edge-connected (kEC) if it remains connected after the removal of an arbitrary subset of $k - 1$ edges.

[3] In [25] a different result is shown with a bi-criteria approximation ratio. However, the result stated here for PAP is given implicitly in [25].

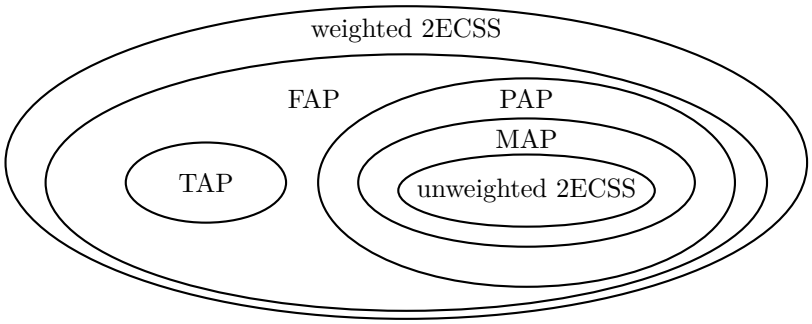


Figure 1 Hierarchical relation between weighted/unweighted 2ECSS, FAP, TAP, PAP, and MAP.

both the reduction to structured instances and the new relaxation 2ECPC as well as the corresponding packing problem TPP may be of independent interest for other connectivity augmentation problems.

1.2 Previous and Related Work

FAP generalizes several well-known network design problems, many of which have seen improvements beyond the 2-approximation barrier, see Figure 1. One important special case is the Tree Augmentation Problem (TAP), where F is a tree. Over the past decades, a sequence of results [1, 9, 10, 11, 14, 17, 19, 20, 26, 32, 36, 37, 38, 39] has led to a 1.393-approximation by Cecchetto, Traub, and Zenklusen [6]. The weighted version of TAP has also been extensively studied, culminating in a $1.5 + \varepsilon$ approximation by Traub and Zenklusen [42], which even holds for the more general connectivity augmentation problem. In the connectivity augmentation problem we are given a k-edge-connected graph G and additional links L and the task is to compute a set of links $X \subseteq L$ of minimum size such that $G \cup X$ is $(k+1)$-edge-connected. Before, the barrier of 2 for connectivity augmentation was breached by Byrka, Grandoni, and Jabal Ameli [5].

Another relevant special case of FAP and PAP is the unweighted 2-edge-connected spanning subgraph problem (unweighted 2ECSS), where $F = \emptyset$. Several works [12, 22, 30, 33, 35, 41] have led to better-than-2 approximations. Bosch-Calvo, Garg, Grandoni, Hommelsheim, Jabal Ameli, and Lindermayr [3] obtained a $\frac{5}{4}$-approximation, which was recently improved slightly by Hommelsheim, Lindermayr, and Liu [29].

The Matching Augmentation Problem (MAP) is the special case of PAP in which F is a matching, i.e., each path consists of a single edge. Cheriyan, Dippel, Grandoni, Khan, and Narayan introduced this problem and gave a $\frac{7}{4}$-approximation [7]. Cheriyan, Cummings, Dippel, and Zhu [8] improved this to $\frac{5}{3}$ and Garg, Hommelsheim, and Megow improved this further to $\frac{13}{8}$ [23]. An LP-based approximation algorithm for MAP with worse guarantee, yet simpler analysis has been developed by Bamas, Drygala, and Svensson [2].

Beyond edge connectivity, many of these problems have natural vertex connectivity counterparts, where the objective is to compute a 2-vertex connected spanning subgraph, ensuring connectivity despite a single vertex failure. For the vertex-connected version of unweighted 2ECSS, several algorithms obtain better-than-2 approximations [13, 24, 27, 33]. The current best-known result is a $\frac{4}{3}$-approximation by Bosch-Calvo, Grandoni, and Jabal Ameli [4].

1.3 Notation

We assume that all instances of optimization problems studied in this paper admit a feasible solution, which can be tested in polynomial time. Throughout this paper, we consider an instance of PAP given by $G = (V, E(\mathcal{P}) \cup L)$, where $\mathcal{P}$ is a collection of disjoint paths, $E(\mathcal{P})$ denotes their edge set, and $L \subseteq \binom{V}{2}$ is a set of additional links. We define $E = E(\mathcal{P}) \cup L$ as the set of edges of G and set $p_{\max} := \max_{P \in \mathcal{P}} |E(P)|$ to be the maximum length of a path in $\mathcal{P}$. For $\mathcal{P}' \subseteq \mathcal{P}$ we write $V^{\text{end}}(\mathcal{P}')$ to denote the set of endpoints (endvertices) of all paths in $\mathcal{P}'$, i.e., vertices that have degree at most 1 w.r.t. $E(\mathcal{P}')$. We define $V^{\text{end}} := V^{\text{end}}(\mathcal{P})$. Every vertex of $V(G)$ that is not an endvertex has degree 2 w.r.t. $E(\mathcal{P})$; such vertices are called interior vertices (of some path $P \in \mathcal{P}$). We further define the following subsets of links: $L^{\text{end}} \subseteq L$ is the set of links uv such that $u, v \in V^{\text{end}}$. We use $L^{\text{end,in}} \subseteq L^{\text{end}}$ to denote the set of links of L^{end} whose endpoints belong to the same path, while $L^{\text{end,out}} = L^{\text{end}} \setminus L^{\text{end,in}}$ contains links of L^{end} that connect different paths. For a subgraph H of G we write $L(H)$ to denote the set of links contained in H. For $P \in \mathcal{P}$ and $u, v \in V(P)$, let P^{uv} be the path from u to v on P. We denote by $\text{OPT}(G)$ some optimal solution of G and by $\text{opt}(G)$ the cost of an optimal solution, i.e., the number of links in $\text{OPT}(G)$. If G is clear from context, we simply write OPT and opt, respectively.

Further, we mainly use standard graph notation. Given a set of vertices $U \subseteq V$, we let $G|U$ denote the multi-graph obtained from G by contracting U to a single vertex. For a subgraph H of G we simply write $G|H$ instead of $G|V(H)$. More generally, given a set $\mathcal{T} = \{T_1, \dots, T_k\}$ of vertex-disjoint sets $T_1, \dots, T_k \subseteq V(G)$, $G|\{T_1, \dots, T_k\}$ (or $G|\mathcal{T}$ in short) denotes the graph obtained by contracting each T_i into a single vertex. A set $X \subseteq V$ is a separator if $G \setminus X$ is disconnected. For $V' \subseteq V(G)$ we define $N_G(V')$ as the neighborhood of V' in G and $\delta(V')$ refers to the edge set with exactly one endpoint in V'. A bridge in G is an edge whose removal from G increases the number of connected components. A subgraph H is a 2-edge cover of G if $|\delta_H(v)| \geq 2$ for each $v \in V(G)$.

2 Overview of Our Approach

In this section, we provide an overview of our algorithmic approach to prove Theorem 1. At a high level, we follow the approach recently introduced for FAP and PAP [25], unweighted 2ECSS [3, 22, 29] and MAP [7, 8, 23]. Our approach has four main steps.

1. Reduction to Structured Instances. The first step is a reduction to structured instances. Specifically, we show that for any $\alpha \geq \frac{11}{6}$ and any small constant $\varepsilon > 0$, if an α-approximation exists for structured instances of bounded-length PAP, then an $(\alpha + \varepsilon)$-approximation exists for arbitrary bounded-length instances of PAP. These structured instances possess key properties that enable and simplify our subsequent algorithmic steps. This step has not been utilized in the previous work on FAP and PAP, except for a reduction to PAP without isolated vertices in [25]. We remark that this is the only place where we require the length of the paths in the instance of PAP to be bounded.

2. Computing a Low-Cost Starting Solution. The second step includes one of our key innovations. Here, we compute a *low-cost* starting solution $S \subseteq L$ such that the graph $H = (V, E(\mathcal{P}) \cup S)$ is not necessarily feasible but has important structural properties. In later steps, we will turn this starting solution into a feasible solution. To guide this process, we introduce a credit scheme that assigns to any spanning subgraph H' of G credits to vertices, edges, blocks, and components of H'. The total sum of credits of H' is denoted

by credits(H'), and we define cost(H') = $|H' \cap L|$ + credits(H'), i.e., the number of links in H' plus the total sum of credits assigned to H'. This scheme helps to maintain a cost measure that accounts not only for the number of selected links but also for structural penalties. Roughly speaking, the number of credits assigned to a solution reflects its distance to a feasible solution. To compute the starting solution, we introduce a novel relaxation of PAP, which we use as a lower bound on opt and which is related to 2-edge covers. Instead of computing a solution to the relaxation directly, we consider a closely related packing problem, where the relationship between the packing problem and the relaxation of PAP is similar to the relationship between 2-matchings and 2-edge covers[4]. Although it turns out that the packing problem and the relaxation are still NP-hard (see full version [28]), we can compute a solution S to the packing problem such that $H_0 = (V, E(\mathcal{P}) \cup S)$ satisfies cost(H_0) $\leq 1.9412 \cdot$ opt. In fact, we compute up to $|\mathcal{P}| + 2$ many solutions H and output the one minimizing cost(H). We analyze the cost of this solution using a factor-revealing LP. This (most likely infeasible) solution is used as a starting solution.

In the remainder, we step-by-step turn H_0 into solutions $H_1, H_2, \ldots$ that are "closer to being feasible", while always keeping the invariant that the new solution H_{i+1} satisfies cost(H_{i+1}) $\leq$ cost(H_i). Since the credits are non-negative and the starting solution satisfies cost(H_0) $\leq 1.9412 \cdot$ opt, eventually we will have a feasible solution using at most $1.9412 \cdot$ opt many links.

There are two reasons why a solution H is not yet feasible: 1) there is a bridge in H, and 2) there are several connected components in H. The next two steps deal with these reasons for infeasibility.

3. Bridge-Covering. The third step, called bridge-covering, ensures that the intermediate solution H' is bridgeless. Our goal is to add (and sometimes remove) links to (from) H such that the resulting graph H' is bridgeless, while keeping the invariant cost(H') $\leq$ cost(H). At this stage, we heavily exploit the fact that G is structured, which allows us to prove the existence of cheap augmenting link sets needed to decrease the number of bridges.

4. Gluing. In the fourth and final step, called gluing, we transform H into a feasible solution, i.e., a 2-edge-connected spanning subgraph. At this point, H consists of multiple 2-edge-connected components that must be merged into a single 2-edge-connected component. We achieve this by iteratively adding well-chosen cycles in the *component graph*, ensuring that each step maintains feasibility and preserves the approximation guarantee. The component graph $G^{\text{comp}}(H)$ arises from G by contracting each component of H to a single vertex. To prove the existence of such cycles we again heavily exploit that G is structured.

The remainder of this section provides a more detailed overview of these four steps, along with lemmas formalizing each transformation. The subsequent sections are dedicated to the individual steps, leading to the proof of Theorem 1, which follows from the four main lemmas established before.

3 Reduction to Structured Instances

Unlike previous algorithms for FAP and PAP [25], we introduce a reduction to structured instances, which plays a crucial role in our approach. Informally, a structured instance is one that avoids certain degenerate configurations that would complicate the design of good

[4] For a graph $G = (V, E)$ a 2-matching $M \subseteq E$ satisfies $\delta_M(v) \leq 2$ for all $v \in V$. A classical result states that one can compute a minimum 2-edge cover by computing a maximum 2-matching and greedily adding edges to each vertex $v \in V$ with $\delta_M(v) < 2$; see [40] for more details.

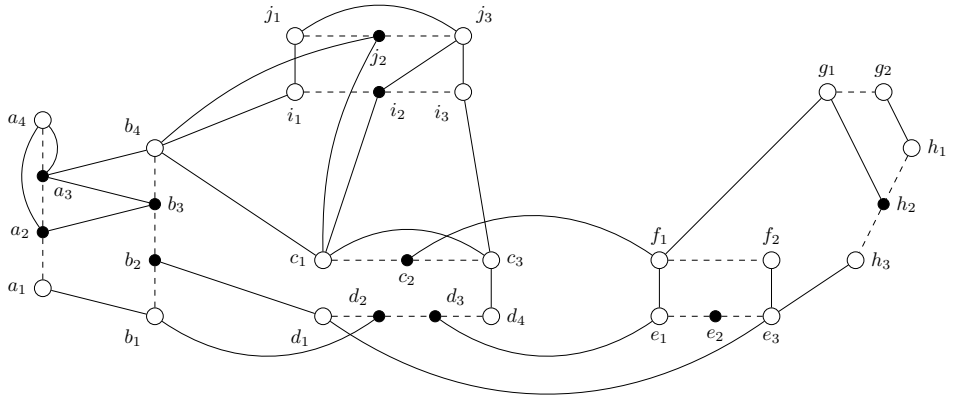


Figure 2 Non-structured PAP instance. Dashed edges are in $E(\mathcal{P})$, solid edges in L. Interior vertices of paths from $\mathcal{P}$ are filled, endvertices of paths from $\mathcal{P}$ are empty. The cycle $a_2a_3a_4a_2$ is a 1-contractible subgraph, the path $P = b_1b_2b_3b_4 \in \mathcal{P}$ is a path-separator, the path $c_1c_2c_3d_4d_3d_2d_1$ is a P^2-separator, the cycle $e_1e_2e_3f_2f_1e_1$ is a C^2-separator, and the path $j_1j_2j_3$ is a degenerate path.

approximation algorithms in our approach. We formally define structured instances and establish that solving bounded-length PAP on structured instances suffices to achieve a good approximation algorithm for general bounded-length PAP. We prove that for all $\alpha \geq \frac{11}{6}$ and any small constant $\varepsilon > 0$, if there exists a polynomial-time α-approximation for bounded-length PAP on structured instances, then there is a polynomial-time $(\alpha+\varepsilon)$-approximation for arbitrary instances of bounded-length PAP. This guarantees that improvements on structured instances directly translate into improvements for the general case.

To define structured instances, we introduce several graph properties, see Figure 2. First, structured instances do not contain contractible subgraphs. Roughly speaking, a subgraph is (strongly) α-contractible if at least a $\frac{1}{\alpha}$ fraction of its links must appear in any feasible solution, allowing us to safely contract it without a loss in the approximation guarantee. We ensure that such subgraphs can be identified and handled efficiently. Formally, we define α-contractible subgraphs as follows.

▶ **Definition 2** (α-contractible subgraph). *Let $\alpha \geq 1$ and $t \geq 1$ be fixed constants. Given a 2-edge-connected graph G, a collection $\mathcal{H}$ of vertex-disjoint 2-edge-connected subgraphs $H_1, H_2, \ldots, H_k$ of G, such that $G|\mathcal{H}$ is also an instance of* PAP*, is called* weakly (α, t, k)-contractible *if $1 \leq |L(H_i)| \leq t$ for every $i \in [k]$ and there exists some optimal solution that contains at least $\frac{1}{\alpha}|\bigcup_{i\in[k]} L(H_i)|$ links from $\bigcup_{i\in[k]} L(G[V(H_i)])$. Furthermore, $\mathcal{H}$ is* strongly (α, t, k)*-contractible if additionally every 2EC spanning subgraph of G contains at least $\frac{1}{\alpha}|\bigcup_{i\in[k]} L(H_i)|$ links from $\bigcup_{i\in[k]} L(G[V(H_i)])$.*

Every strongly (α, t, k)-contractible subgraph is also a weakly (α, t, k)-contractible subgraph. We say that $\mathcal{H}$ is an (α, t, k)-contractible subgraph if it is a weakly or strongly (α, t, k)-contractible subgraph. If α, t, k are clear from the context, we simply refer to α-contractible subgraphs or even contractible subgraphs. A typical example of a strongly contractible subgraph occurs if an endvertex v of some path $P \in \mathcal{P}$ is only adjacent to vertices on P: If u is the neighbor of v on P furthest away from v, then $(V(P^{uv}), P^{uv} \cup \{uv\})$ is strongly $(1, 1, 1)$-contractible, since $(V(P^{uv}), P^{uv} \cup \{uv\})$ is 2-edge-connected and every feasible solution must contain a link connecting v and $V(P^{uv})$. This is summarized in Property (P2) of Definition 4; see also Figure 2.

Second, structured instances avoid certain small separators. Specifically, we prevent the existence of path-separators, P^2-separators, and C^2-separators, which are roughly defined as follows (leaving out some technicalities, see full version [28]). A path-separator is a path $P \in \mathcal{P}$ that is a separator, i.e., $G \setminus V(P)$ is not connected. A P^2-separator is a separator that consists of two paths $P, P' \in \mathcal{P}$ such that there is a link in $L^{\text{end,out}}$ connecting one endvertex of P and one of P'. Note that $E(P) \cup E(P') \cup \{e\}$ is a simple path on $V(P) \cup V(P')$. Similarly, a C^2-separator is a separator that consists of two paths $P, P' \in \mathcal{P}$ such that there are two links in $e, e' \in L^{\text{end,out}}$ such that $E(P) \cup E(P') \cup \{e, e'\}$ is a cycle. Note that a C^2-separator is also a P^2-separator, but the additional edge required for C^2-separators allows us to obtain stronger results due to technical reasons omitted here. If an instance contains such a separator, we decompose it into smaller subinstances, solve each recursively, and merge their solutions while maintaining the desired approximation guarantee. Formal definitions are given in the full version [28]. We remark that we only require the length of the paths in the instance of PAP to be bounded for the reduction to avoid path-separators and P^2-separators.

A final obstacle to obtain structured instances is the presence of too many degenerate paths, which are paths $P \in \mathcal{P}$ with limited connectivity to the rest of the graph. When the fraction of degenerate paths exceeds the threshold $\varepsilon|\mathcal{P}|$, we identify certain well-connected subgraphs that we can contract and then solve the resulting subinstances recursively.

▶ **Definition 3** (Degenerate Path). *Let $P, P' \in \mathcal{P}$ be two distinct paths with endpoints u', v' and u, v, respectively, such that $u'u \in L$, $v'v \in L$ and $u'v' \in L$. We say that P' is a degenerate path if $N(u'), N(v') \subseteq V(P') \cup V(P)$.*

Using these definitions, we define structured instances.

▶ **Definition 4** ((α, ε)-structured Instance). *We call an instance $G = (V, \mathcal{P} \cup L)$ of bounded-length PAP (α, ε)-structured if the following properties are satisfied:*

(P0) $\text{opt}(G) > \frac{3}{\varepsilon}$.
(P1) *G does not contain strongly $(\alpha, t, 1)$-contractible subgraphs with $t \leq 10$ many links.*
(P2) *Every endpoint of a path $P \in \mathcal{P}$ has a link to some other path $P' \in \mathcal{P}$, $P' \neq P$.*
(P3) *G does not contain path-separators.*
(P4) *G does not contain P^2-separators.*
(P5) *G does not contain C^2-separators.*
(P6) *G contains at most $\varepsilon \cdot |\mathcal{P}|$ many degenerate paths.*
(P7) *Every vertex $v \in V$ is on some path $P \in \mathcal{P}$ that contains at least 2 vertices.*

If α and ε are clear from the context, we simply say G is structured. In the subsequent subsections we outline how these properties are useful to obtain good approximation algorithms. Our main lemma in this section is the following.

▶ **Lemma 5.** *For all $\alpha \geq \frac{11}{6}$ and constant $\varepsilon \in \left(0, \frac{1}{12}\right]$, if there exists a polynomial-time α-approximation algorithm for bounded-length* PAP *on (α, ε)-structured instances, then there exists a polynomial-time $(\alpha + \varepsilon)$-approximation algorithm for bounded-length* PAP *on arbitrary instances.*

We give a rough overview on how we obtain Lemma 5. Assume that G is not structured. In this overview, we focus only on three obstacles that prevent G from being structured: 1) G contains a contractible subgraph, 2) G contains a forbidden separator, or 3) G contains more than $\varepsilon|\mathcal{P}|$ many degenerate paths. If G contains one of these subgraphs, we can decompose G into at most 2 smaller subinstances, solve each recursively, and merge their solutions while maintaining the approximation guarantee. We outline our approach for each of these three cases.

Property (P1) is not met. We can safely contract each component of an α-contractible subgraph $\mathcal{H}$ into a single vertex and work with $G|\mathcal{H}$, as long as we only aim for an α-approximation. Indeed, any α-approximate solution for $G|\mathcal{H}$ can be turned into an α-approximate solution for G by adding the links of $\mathcal{H}$. In general, it is not straightforward to argue that a 2EC subgraph is weakly α-contractible. However, under certain conditions we can argue that this is indeed the case. We will show that we can find strongly $(\alpha, t, 1)$-contractible subgraphs in polynomial time if t is constant, i.e., if there is only one component and there are only a constant number of links in the contractible subgraph. Hence, after our preprocessing, we can assume that G does not contain any strongly $(\alpha, t, 1)$-contractible subgraphs for some constant t (here $t = 10$ suffices).

Property (P3), (P4), or (P5) is not met. Assume that G contains a forbidden separator X, resulting in $G \setminus X$ to decompose into two disjoint vertex sets V_1 and V_2 and there are no edges between V_1 and V_2. Here, we construct two sub-instances $G_1 = (G \setminus V_2)|X$ and $G_2 = (G \setminus V_1)|X$ arising from G by removing either V_1 or V_2 and contracting X into a single vertex. We recursively solve these sub-instances approximately and combine both solutions while preserving the desired approximation guarantee. By introducing a proper notion of *size* of an instance, one can ensure that the sum of the sizes of the two sub-instances is smaller than the size of the original instance and hence this step runs in polynomial time.

Property (P6) is not met. One of the main innovations in our reduction to structured graphs compared to previous work deals with degenerate paths. For degenerate paths, we can not do the same strategy as for separators and instead do the following: Let $\mathcal{P}' = \{P'_1, \dots, P'_\ell\}$ be a set of degenerate paths. We define two collections $\mathcal{K}$ and $\mathcal{C}$ of ℓ vertex-disjoint 2-edge-connected subgraphs, each subsuming $V(\mathcal{P}')$. We show that either $\mathcal{C}$ is weakly $(\alpha, 2, \ell)$-contractible or $\mathcal{K}$ is weakly $(\alpha, 1, \ell)$-contractible for any $\alpha \geq \frac{11}{6}$. Hence, either an α-approximate solution for $G|\mathcal{K}$ or an α-approximate solution for $G|\mathcal{C}$ can be turned into an α-approximate solution for G by adding the edges contained in $\mathcal{K}$ or $\mathcal{C}$, respectively.

Unfortunately, in general this recursive procedure leads to an exponential running time if $|\mathcal{P}'|$ is small: We need to recursively solve the two sub-instances $G|\mathcal{K}$ and $G|\mathcal{C}$ that both may have roughly the same size as the original instance. Since we might need to apply this reduction again in later steps in our reduction to structured graphs, this procedure can result in an exponential running time. However, we circumvent this by only applying the above procedure if some constant fraction ε of the paths in $\mathcal{P}$ is degenerate, and hence each sub-instance is significantly smaller than the original one, which overall results in a polynomial running time.

In light of Lemma 5, it remains to give a 1.9412-approximation for bounded-length PAP on $(\frac{11}{6}, \varepsilon)$-structured instances. Hence, for the remaining steps, we assume the instance $G = (V, E(\mathcal{P}) \cup L)$ is a $(\frac{11}{6}, \varepsilon)$-structured instance of bounded-length PAP. In fact, for the remainder of this paper we only require the instance to be a $(\frac{11}{6}, \varepsilon)$-structured instance of PAP; the requirement of bounded-length was only needed for the reduction to structured graphs.

4 Credit Scheme, Relaxation, and Starting Solution

Throughout our algorithm, we maintain a temporary solution $S \subseteq L$, which is not necessarily a feasible solution. To track progress and guide modifications, we assign certain credits to parts of the current graph $H = (V, E(\mathcal{P}) \cup S)$. These credits are used to ensure that

every transformation preserves an approximation bound while gradually converting H into a feasible solution. We first define key structural components and introduce a cost function that incorporates both the number of selected links and assigned credits.

We use the following notation of blocks from related works on unweighted 2ECSS [22, 3, 29] and MAP [7, 8, 23]. A block is a maximal 2-edge-connected subgraph containing at least 2 vertices. A block B is called *simple* if there is some $P \in \mathcal{P}$ such that $V(B) \subseteq V(P)$, it is called *small* if there are two distinct paths $P_1, P_2 \in \mathcal{P}$ such that $V(B) = V(P_1) \cup V(P_2)$, and *large* if there are three distinct paths $P_1, P_2, P_3 \in \mathcal{P}$ such that $V(P_1) \cup V(P_2) \cup V(P_3) \subseteq V(B)$. A connected component that is 2EC is called a 2EC *component*. We call 2EC components small if they contain exactly 2 links and large if they contain at least 3 links.

Let H be a spanning subgraph of G, and let C be a connected component of H. A component is called *complex* if it contains bridges. For any such C, we define the graphs G_C and H_C as follows: G_C is the graph obtained from G by contracting every connected component of H except for C, as well as every block of C. H_C is the corresponding contraction of H. The structure of H_C consists of a tree T_C along with singleton vertices. A vertex in G_C or H_C that corresponds to a block in H is called a *block vertex*. If a block vertex corresponds to a simple block, it is called a *simple block vertex*. We identify the edges/links in G and the corresponding edges/links in G_C and H_C, resp.

A component C is called *trivial* if it consists of a single path $P \in \mathcal{P}$ (i.e., $V(C) = V(P)$). Otherwise, it is called *non-trivial*. A vertex $v \in V$ is called *lonely* if it does not belong to any 2-edge-connected component or block of H. For two paths $P, P' \in P$ with endpoints u, v and u', v', resp., we say that $uu' \in L$ is an *expensive link* if $N_G(v) \subseteq (V(P) \cup V(P')) \setminus \{v'\}$. If additionally $uu' \in L(H)$ is a bridge in H and v is a lonely leaf in H, then v is called an *expensive leaf vertex*.

For a given $H = (V, E(\mathcal{P}) \cup S)$, we assign credits as follows.

- **(A)** Every lonely vertex v that is a leaf of some complex component C receives 1 credit.
  - **(i)** It receives an additional $\frac{1}{4}$ credit if v is an expensive leaf vertex.
  - **(ii)** It receives an additional $\frac{1}{4}$ credit if there is a path in H_C from v to a simple block vertex of C without passing through a non-simple block vertex.
- **(B)** Every bridge $\ell \in S$ (i.e., a link that is not part of any 2-edge-connected component of H) receives $\frac{3}{4}$ credits.
- **(C)** Every complex component receives 1 credit.
- **(D)** Every non-simple 2-edge-connected block receives 1 credit.
- **(E)** Every 2-edge-connected component receives:
  - 2 credits if it is large or contains a degenerate path.
  - $\frac{3}{2}$ credits if it is small (i.e., has exactly 2 links) and does not contain a degenerate path.

For any spanning subgraph $H = (V, E(\mathcal{P}) \cup S)$, we define the total credit as credits(H), which is the sum of all assigned credits under these rules. We define cost$(H) :=$ credits$(H) + |S|$.

As an example, consider the right solution in Figure 3. There are three connected components: (1) The 2EC component containing v_1, which receives a credit of 2 (Rule (E)). (2) The complex component containing v_7. This component receives a credit of 1 (Rule (C)), the bridge u_7u_8 receives a credit of $\frac{3}{4}$ (Rule (B)), and the lonely leaf vertices u_7, u_8 receive a credit of 1 each (Rule (A)). (3) The complex component containing v_6. This component receives a credit of 1 (Rule (C)), the bridges $u_6w_3, w_3w_4, w_4u_5, v_5u_9, v_9w_{11}, w_{11}v_{10}$ receive a credit of $\frac{3}{4}$ (Rule (B)) each, and the lonely leaf vertices v_6, u_{10} receive a credit of 1 each (Rule (A)) (potentially plus $\frac{1}{4}$ if Rule (A) (i) applies).

To prove the approximation guarantee, we show that we maintain the following invariants.

▶ **Invariant 6.** $\text{cost}(H) \leq 1.9412 \cdot \text{opt}(G)$.

▶ **Invariant 7.** *Every lonely vertex of H has degree at most 2 in H. Every simple 2EC block B is part of some complex component C and has degree exactly 2 in H_C. Furthermore, for any complex component C, no two simple block vertices of C are adjacent to each other in H_C.*

▶ **Invariant 8.** *For every non-simple 2EC block B of H there are two distinct paths $P, P' \in \mathcal{P}$ such that $V(P) \cup V(P') \subseteq V(B)$.*

Our goal is to design a polynomial-time algorithm that computes a solution $S \subseteq L$ such that $H = (V, E(\mathcal{P}) \cup S)$ satisfies all invariants above. This is formalized in the following lemma.

▶ **Lemma 9.** *Given a structured graph, in polynomial time we can compute a set of links $S \subseteq L$ such that $H = (V, E(\mathcal{P}) \cup S)$ satisfies the Invariants 6-8.*

Next, we outline how to compute such a solution. The approaches in [3, 22, 7, 8, 23] for special cases of PAP, such as unweighted 2ECSS and MAP, use a relaxation of the problem, for which an optimum solution can be computed in polynomial time and a solution serves as a good starting solution. In particular, these prior works used a minimum 2-edge cover as a relaxation. For *structured instances* of these special cases of PAP, optimum 2-edge covers are usually not too far away from an optimum solution of the original problem (i.e., unweighted 2ECSS or MAP), and hence they serve as a good lower bound. However, for PAP, a 2-edge cover can be far away from an optimum solution. For example, if the 2-edge cover of an instance of PAP only contains edges of $E(\mathcal{P})$ and $L^{\text{end,in}}$, then this edge set only 2-edge-connects the paths individually, but it does not help to establish overall 2-edge-connectivity. This motivates us to define a more suitable relaxation for PAP, called 2-Edge Cover with Path Constraints (2ECPC).

4.1 A relaxation of PAP: The 2-Edge Cover with Path Constraints

The input consists of a structured instance of PAP where each path $P_i \in \mathcal{P}$ has exactly three vertices: two endpoints $u_i, v_i \in V^{\text{end}}(P_i)$ and an interior vertex w_i. The goal is to compute a minimum-cardinality link set $X \subseteq L$ such that $E(\mathcal{P}) \cup X$ is a 2-edge cover with the additional constraint that each path P_i is incident to at least two outgoing links in X. More formally, X has to satisfy the following:

- Every vertex $u \in V^{\text{end}}(P_i)$ is incident to at least one link in X, i.e., $|\delta_X(u)| \geq 1$.
- Each path $P_i \in P$ has at least two outgoing edges in X, i.e., $|\delta_X(P_i)| \geq 2$.

One can easily obtain an instance of 2ECPC from an instance of PAP by doing the following for each path: If the path contains at least 4 vertices, we contract all internal vertices to a single vertex, and if the path contains 2 vertices, we add a dummy vertex as an interior vertex. Recall that for structured instances, every vertex is part of some path $P \in \mathcal{P}$ containing at least 2 vertices (Property (P7)). There is a one-to-one correspondence between links in the graph of the instance of PAP and links in the graph of the instance of 2ECPC. One can verify that any feasible solution to the PAP instance is also feasible for its corresponding 2ECPC instance; hence 2ECPC is indeed a relaxation for PAP. Instead of computing an (approximate) solution to 2ECPC, we consider a packing problem, which we call the Track Packing Problem (TPP).

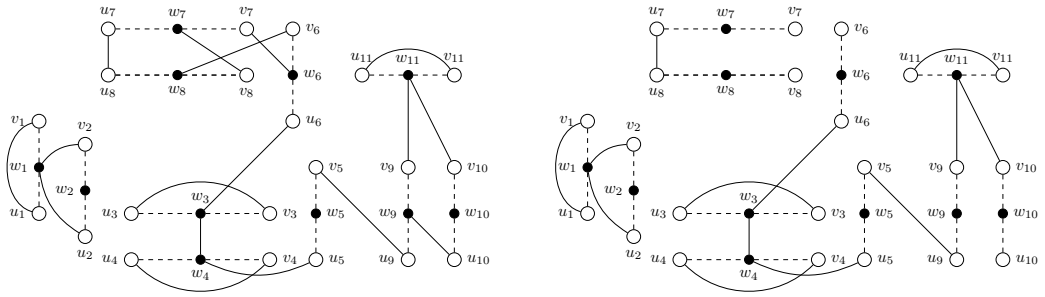


Figure 3 Dashed edges are in $E(\mathcal{P})$, solid edges are links of L. Filled vertices are interior vertices of paths from $\mathcal{P}$, empty vertices are endvertices of paths in $\mathcal{P}$. Left: Feasible solution Y for 2ECPC, since $E(\mathcal{P}) \cup Y$ is a 2-edge cover with $\delta_Y(P) \geq 2$ for each $P \in \mathcal{P}$. Right: Solution $X = \{T_1, T_2, T_3, T_4, T_5\}$ for the corresponding TPP instance with two tracks containing 1 link ($T_1 = \{u_7u_8\}$ and $T_2 = \{v_5u_9\}$), two tracks containing 3 links ($T_3 = \{u_2w_1, u_1v_1, w_1v_2\}$ and $T_4 = \{v_9w_{11}, u_{11}v_{11}, w_{11}v_{10}\}$) and one track containing 5 links ($T_5 = \{u_5w_4, u_4v_4, w_4w_3, u_3v_3, w_3u_6\}$). Note that $|Y| = 17 = 2|\mathcal{P}| - 5 = 2|\mathcal{P}| - |X|$.

We denote by OPT and OPT_C the optimum solutions for the PAP and the 2ECPC instance, respectively, and define opt and opt_C to be their objective values. It is easy to verify that any feasible solution to the PAP instance is also feasible for its corresponding 2ECPC instance, and hence 2ECPC is indeed a relaxation for PAP. We summarize this as follows.

▶ **Proposition 10.** *Let G be a structured instance of* PAP. *Then* $\text{opt} \geq \text{opt}_C$.

It is easy to see that $|\mathcal{P}| \leq \text{opt}_C \leq 2|\mathcal{P}|$: Observe that since the input graph G is structured, each endpoint u of some path $P \in \mathcal{P}$ is incident to a link $e \in \delta(P)$. Therefore, by picking such an edge for each endpoint u, there is a straightforward solution containing at most $2|\mathcal{P}|$ many links. Furthermore, note that any feasible solution must pick at least $|\mathcal{P}|$ many links. Therefore, we obtain the inequalities above.

Similar to other connectivity augmentation problems, we assume that the instance of 2ECPC is *shadow-complete*. In our setting this means the following: Let $P_i, P_j \in P$ such that $P_i \neq P_j$. If there is a link u_iu_j from an endpoint u_i of P_i to some other endpoint u_j of P_j, then we also add the *weaker* links u_iw_j, w_iu_j and w_iw_j, where w_i, w_j are the interior vertices of the paths P_i and P_j, respectively. Furthermore, if there is a link u_iw_j from an endpoint u_i of P_i to the interior vertex w_j of P_j, then we also add the *weaker* link w_iw_j, where w_i is the interior vertex of P_i. These weaker links are then also called the *shadows* of their original links. An instance that contains all such shadows is called shadow-complete.

It is not hard to see that for any feasible instance of 2ECPC, a feasible solution to the shadow-complete instance can be turned into a feasible solution to the original instance of the same cost.

▶ **Lemma 11.** *For any feasible instance of* 2ECPC *arising from a structured graph, a feasible solution Y to the shadow-complete instance can be turned into a feasible solution Y' to the original instance such that $|Y'| \leq |Y|$.*

Hence, from now on we assume that our instance of 2ECPC is shadow-complete.

4.2 The Track Packing Problem

TPP is inspired by the relationship between 2-matchings and 2-edge covers, where the covering constraints are expressed as a packing problem. Formally, we construct an instance of TPP from a shadow-complete instance of 2ECPC (see full version [28] for formal definitions). For some link set $A \subseteq L$ let $V^{\text{end}}(A) \coloneqq V^{\text{end}}(\mathcal{P}) \cap V(A)$ be the set of vertices that are endpoints of some path and are incident to some link in A. A *track* $T \subseteq L$ with $T = Q(T) \cup I(T)$ is a subset of links such that the following conditions hold.

- $Q(T)$ is a simple path in G of length at least one, where the start vertex u and the endvertex v of $Q(T)$ are endpoints of some paths $P, P' \in \mathcal{P}$ (not necessarily $P \neq P'$) such that $u \neq v$,
- any interior vertex of $Q(T)$, i.e., a vertex from $V(Q(T)) \setminus \{u, v\}$, is the interior vertex of some path $P \in \mathcal{P}$,
- $I(T) \subseteq L$ contains exactly the following set of links: for each interior vertex $w_i \in V(Q(T)) \setminus \{u, v\}$, which is an interior vertex of some path $P_i \in \mathcal{P}$, there is a link $u_i v_i \in I(T)$, where $u_i, v_i \in V^{\text{end}}(P_i)$ are the endvertices of P_i, and
- if $|T| = 1$, i.e., if $|Q(T)| = 1$ and $|I(T)| = 0$, then the start vertex u of $Q(T)$ and the endvertex v of $Q(T)$ belong to distinct paths.

For any track T, we refer to $Q(T)$ and $I(T)$ as above. Observe that the number of links in a track is odd, since $|Q(T)| = |I(T)| + 1$. Note that a track of length 1 is simply a link in $L^{\text{end,out}}$, i.e., a link between two endvertices of two distinct paths.

The instance of TPP is now defined as follows. The input is a (shadow-complete) instance of 2ECPC of some structured graph G. Let $\mathcal{T}$ be the set of all tracks in G. The task is to select a maximum number of disjoint tracks from $\mathcal{T}$, where two tracks T, T' are disjoint if $V(T) \cap V(T') = \emptyset$.

In the following we establish a relationship between solutions of 2ECPC and its corresponding TPP instance. This connection is similar to the connection between 2-matchings and 2-edge covers. In particular, an optimum solution to one of the problems can be turned into an optimum solution to the other problem in polynomial time. We note that the additional condition required in Lemma 12 and Lemma 13 is always satisfied for structured graphs. The relationship between 2ECPC and TPP is highlighted in Figure 3.

▶ **Lemma 12.** *Given a solution X for a shadow-complete instance of* TPP*, in which each endvertex of a path is connected to a vertex of a different path, we can compute in polynomial time a solution Y to the corresponding* 2ECPC *instance of value $2|\mathcal{P}| - |X|$.*

▶ **Lemma 13.** *Consider a shadow-complete instance I of* 2ECPC*, in which each endvertex of a path is connected to a vertex of a different path. Given an inclusion-wise minimal solution Y to I such that $|Y| \leq 2|\mathcal{P}|$, we can compute in polynomial time a solution X to the corresponding* TPP *instance of value at least $2|\mathcal{P}| - |Y|$.*

We also note that TPP is NP-hard by a reduction from the NP-hard P_2-Packing problem [34]. In the P_2-Packing problem we are given a graph G and the task is to select a maximum number of vertex-disjoint P_2's, i.e., paths that contain exactly 2 edges.

▶ **Proposition 14.** *The Track Packing Problem is* NP*-hard, even on shadow-complete instances, in which each endvertex of a path is connected to a vertex of a different path.*

Note that due to the above relationship between 2ECPC and TPP established in Lemma 12 and Lemma 13, Proposition 14 also implies NP-hardness for 2ECPC.

▶ **Proposition 15.** *The 2-Edge Cover with Path Constraints is* NP*-hard.*

Despite this hardness, in the remaining part we show how to compute a solution to TPP which serves as a good starting solution for our algorithmic framework.

4.3 Computing a good starting solution

Our goal is to compute a solution to TPP. By its close relationship to 2ECPC, we can bound the cost of this solution in terms of opt_C, the optimum solution value for the instance of 2ECPC, and since 2ECPC is a relaxation of PAP, we can in turn bound its cost in terms of opt. We do not compute an approximate solution to TPP relative to its optimum solution value; instead we do the following: For a solution X to TPP we compute $H_X = (V, E(\mathcal{P}) \cup (X \cap L))$, and we want to find a solution X to TPP such that $\text{cost}(H_X) = |X \cap L| + \text{credits}(H_X) \leq 1.9412 \cdot \text{opt}$.

To find such a solution, we view TPP as a Set-Packing problem. Therein, we are given a set of elements U and a set of subsets $\mathcal{S} \subseteq 2^U$ of U and the task is to compute a set $X \subseteq \mathcal{S}$ of maximum cardinality such that for any two sets $S, S' \in X$ we have $S \cap S' = \emptyset$. Set-Packing is APX-hard and the state-of-the-art polynomial-time approximation algorithms for it achieve an approximation factor of $\frac{3}{1+k+\varepsilon}$ for any constant $\varepsilon > 0$, where $k \coloneqq \max_{S \in \mathcal{S}} |S|$ is the maximum size of a set in $\mathcal{S}$ [15, 21]. TPP can be reduced to Set-Packing by viewing each $v \in V^{\text{end}}(\mathcal{P})$ as an element and each track as a set, which exactly contains those vertices $v \in V^{\text{end}}(\mathcal{P})$ to which the track is incident to.

Using such set-packing algorithms, we define three different algorithms that compute sets of disjoint tracks, where the maximum size of a track we consider contains at most 4 vertices of $V^{\text{end}}(\mathcal{P})$. In total, we compute up to $|\mathcal{P}| + 2$ different solutions and output the solution X minimizing $\text{cost}(H_X)$. It can be easily shown that H_X satisfies Invariant 7 and Invariant 8. To show that H_X satisfies Invariant 6, that is, $\text{cost}(H_X) \leq 1.9412 \cdot \text{opt}$, we bound the cost of H_X in terms of variables that depend on the three algorithms and certain parameters of an optimum solution. Here, we heavily exploit the connection between optimum solutions of TPP and 2ECPC from above. We then derive a factor-revealing LP, which proves $\text{cost}(H_X) \leq \rho \cdot \text{opt}$ if the polyhedron is empty for a specific value of ρ. Indeed, for $\rho = 1.9412$ this is the case, implying Invariant 6. This then proves Lemma 9.

Throughout this section we fix an optimum solution OPT_P for the instance of TPP and assume that $\text{OPT}_P = \bigcup_{i=1}^{n} O_i$, where O_i is the set of tracks T in the optimum solution such that $|Q(T)| = i$. We define $\omega_i = |O_i|$. Hence, $|\text{OPT}_P| = \sum_{i=1}^{n} \omega_i$. We additionally let ω_0 be the number of vertices of $V^{\text{end}}(\mathcal{P})$ that are not incident to any track in OPT_P. Since each endvertex of $V^{\text{end}}(\mathcal{P})$ can be incident to at most one track and each track T with $|Q(T)| = i$ is incident to $2i$ distinct endvertices of $V^{\text{end}}(\mathcal{P})$, we have that $\omega_0 + \sum_{i=1}^{n} 2i \cdot \omega_i = 2|\mathcal{P}|$. Finally, let $\omega_1^e \leq \omega_1$ be the number of tracks T contained in OPT_P such that $|Q(T)| = 1$ and T corresponds to an expensive link.

Algorithm A

In Algorithm A we first only consider tracks of size 1, which are just links from $L^{\text{end,out}}$. In the first step we compute a maximum number of disjoint tracks of size 1, i.e., we compute a maximum matching $A_1 \subseteq L^{\text{end,out}}$, which can be done in polynomial time.

For the second step, let R_A be the set of endvertices of $V^{\text{end}}(\mathcal{P})$ that are not incident to some link of A_1. Let I_{R_A} be the resulting set-packing instance where we have one element for each vertex in R_A and consider only those tracks T that are only incident to vertices

from R_A and additionally satisfy $|Q(T)| = 2$ (i.e., $Q(T)$ contains exactly 2 links). We then compute a solution A_2 to I_{R_A} using the $\frac{3}{5+\varepsilon}$-approximation algorithm from [21] and output the solution $S_A = A_1 \cup A_2$.

Algorithm B

Algorithm B is similar to Algorithm A, but we compute a different matching in the first step. We aim at "guessing" the value ω_1^e, i.e., how many tracks of size 1 are contained in the optimum solution OPT_P that correspond to an expensive link. There are at most $|\mathcal{P}|$ many guesses, hence we can simply try all possible values.

Given a guess of ω_1^e, we compute a subset $B_1 \subseteq L^{\mathrm{end,out}}$ of maximum size, such that B_1 contains at most ω_1^e many expensive links. This can be done by guessing the size k of the maximum matching with at most ω_1^e many expensive links and giving each expensive link in $L^{\mathrm{end,out}}$ a cost of 1 and any other link of $L^{\mathrm{end,out}}$ a cost of 0 and computing a minimum-cost matching of size exactly k. The maximum k for which the cost of the solution is at most ω_1^e is then the desired solution. This gives us the edge set B_1, which corresponds to a maximum set of disjoint tracks of length 1 with at most ω_1^e many expensive links. Note that a minimum-cost matching of size exactly k in a weighted graph with n vertices (and non-negative weights) can be found by reducing it to a minimum-cost perfect matching problem by adding $n - 2k$ dummy vertices and connecting each dummy vertex to every original vertex with an edge of cost 0.

The second step in Algorithm B is identical to the second step of Algorithm A: Let R_B be the set of endvertices of $V^{\mathrm{end}}(\mathcal{P})$ that are not incident to some link of B_1. Let I_{R_B} be the resulting set-packing instance where we have one element for each vertex in R_B and consider only those tracks T that are only incident to vertices from R_B and additionally satisfy $|Q(T)| = 2$ (i.e., $Q(T)$ contains exactly 2 links). We then compute a solution B_2 to I_{R_B} using the $\frac{3}{5+\varepsilon}$-approximation algorithm from [21] and output the solution $S_B = B_1 \cup B_2$.

Algorithm C

In the third algorithm, we directly consider the corresponding set-packing problem in which we only consider tracks T with $|Q(T)| \leq 2$ (i.e., tracks T such that $|Q(T)| = 1$ or $|Q(T)| = 2$). We then compute a solution S_C to this set-packing instance using the $\frac{3}{5+\varepsilon}$-approximation algorithm from [21] and output S_C. We set $S_C = C_1 \cup C_2$, where C_i is the set of tracks T such that $|Q(T)| = i$.

Final Algorithm

The final solution is then computed as follows. For a solution $H = (V, E(\mathcal{P}) \cup S)$ we defined $\mathrm{cost}(H) \coloneqq \mathrm{credits}(H) + |S|$. For each guess $i \in \{0, 1, \ldots, |\mathcal{P}|\}$ of ω_1^e in Algorithm B we have a solution $H^i_{\mathrm{AlgB}} = (V, E(\mathcal{P}) \cup S^i_B)$ and for Algorithm A and C we have one solution $H_{\mathrm{AlgA}} = (V, E(\mathcal{P}) \cup S_A)$ and $H_{\mathrm{AlgC}} = (V, E(\mathcal{P}) \cup S_C)$, respectively. For each of the $|\mathcal{P}| + 3$ many solutions, we then compute their respective cost as defined above and output the solution with lowest cost. We let $H = (V, E(\mathcal{P}) \cup S)$ be this final solution.

We note that in fact one of the solutions computed in Algorithm B will be identical to the solution computed in Algorithm A. This is true if the guess of ω_1^e is 0 and hence in Algorithm B, we simply compute a maximum matching of $L^{\mathrm{end,out}}$. However, for notational purposes it will be more convenient to also consider Algorithm A with its solution S_A.

4.4 Bounding the cost of the solution

We now bound $\text{cost}(H)$. We bound the number of links in the optimum solution as follows.

▶ **Lemma 16.** *We have* $\text{opt}_C \geq 2|\mathcal{P}| - \sum_{i\geq 1} \omega_i$.

Proof. By Lemmas 12 and 13 we have that $\text{opt}_C = 2|\mathcal{P}| - \text{opt}_P$. By definition, we have that $\text{opt}_P = \sum_{i\geq 1} \omega_i$ and therefore obtain the result. ◀

Let S_B be the solution from Algorithm B which corresponds to the correct guess of ω_1^e. From now on, we only need this solution from Algorithm B and do not consider the solutions for the other guesses of ω_1^e. Let $S_A = A_1 \cup A_2$, where A_i is the set of tracks T from S_A such that $|Q(T)| = i$ and define $\alpha_i = |A_i|$ for $i = 1, 2$. Similarly, we define $S_B = B_1 \cup B_2$ and $S_C = C_1 \cup C_2$ and set $\beta_i = |B_i|$ and $\gamma_i = |C_i|$ for $i = 1, 2$. Furthermore, let $\alpha_0, \beta_0, \gamma_0$ be the number of vertices from $V^{\text{end}}(\mathcal{P})$ that are not incident to some track of S_A, S_B or S_C, respectively. Note that such vertices correspond to leaves in S_A and S_B, respectively. Additionally, let $\alpha_0^e \leq \alpha_0$, $\beta_0^e \leq \beta_0$, and $\gamma_0^e \leq \gamma_0$ be the number of such leaves as above that are expensive w.r.t. S_A, S_B and S_C, respectively. Finally, for $j = 0, 1, 2$ let α_1^j be the number of tracks $T = uv$ of size 1 in S_A such that exactly j vertices of $\{u, v\}$ intersect some track of size 1 in OPT_P. Similarly, for $j = 0, 1, 2$ we define β_1^j. Note that $\alpha_1 = \alpha_1^0 + \alpha_1^1 + \alpha_1^2$ and $\beta_1 = \beta_1^0 + \beta_1^1 + \beta_1^2$. First, we prove the following bound on the cost of a solution by giving it certain credits.

▶ **Lemma 17.** *Let S be a solution computed by Algorithm A, B, or C. If each track of length 1 receives a credit of $\frac{3}{4}$, each track of length 3 a credit of 2, each lonely leaf vertex a credit of $\frac{3}{2}$ if it is not expensive and $\frac{7}{4}$ if it is expensive, and each degenerate path an additional credit of $\frac{1}{2}$, then we can redistribute the credits such that the credit rules (A)-(E) are satisfied. Furthermore, Invariants 7 and 8 are satisfied.*

Next, we show that also Invariant 6 is satisfied. To do so, we establish the following lemmas.

▶ **Lemma 18.** *If we assign to H_{AlgA}, H_{AlgB} and H_{AlgC} credits according to Lemma 17, then we have* $\text{cost}(H_{\text{AlgA}}) \leq 3|\mathcal{P}| - \frac{5}{4}\alpha_1 - \alpha_2 + \frac{1}{4}\alpha_0^e + \frac{1}{2}\varepsilon'|\mathcal{P}|$, $\text{cost}(H_{\text{AlgB}}) \leq 3|\mathcal{P}| - \frac{5}{4}\beta_1 - \beta_2 + \frac{1}{4}\beta_0^e + \frac{1}{2}\varepsilon'|\mathcal{P}|$ *and* $\text{cost}(H_{\text{AlgC}}) \leq 3|\mathcal{P}| - \frac{5}{4}\gamma_1 - \gamma_2 + \frac{1}{4}\gamma_0^e + \frac{1}{2}\varepsilon'|\mathcal{P}|$, *where $\varepsilon'|\mathcal{P}|$ is the number of degenerate paths in G.*

Next, we obtain some further constraints on the variables we defined earlier.

▶ **Lemma 19.** *The following inequalities are valid:*

1. $\alpha_1 \geq \omega_1, \beta_1, \gamma_1$,
2. $\alpha_1 \geq \omega_1 + \alpha_1^0$,
3. $\alpha_1 \geq \omega_1 + \beta_1^0$,
4. $\beta_1 \geq \omega_1$,
5. $\alpha_1 \geq \alpha_0^e$,
6. $\beta_1 \geq \beta_0^e$,
7. $\gamma_1 \geq \gamma_0^e$,
8. $\omega_0 \geq \beta_0^e$,
9. $\alpha_1 = \alpha_1^0 + \alpha_1^1 + \alpha_1^2$,
10. $\beta_1 = \beta_1^0 + \beta_1^1 + \beta_1^2$,
11. $\alpha_2 \geq \frac{3}{5+\varepsilon} \cdot (\omega_2 - 2\alpha_1^0 - \alpha_1^1)$,
12. $\beta_2 \geq \frac{3}{5+\varepsilon} \cdot (\omega_2 - 2\beta_1^0 - \beta_1^1)$, *and*
13. $\gamma_1 + \gamma_2 \geq \frac{3}{5+\varepsilon} \cdot (\omega_1 + \omega_2)$.

Recall that $H = (V, E(\mathcal{P}) \cup S)$, where S is the best solution output by Algorithms A, B, and C w.r.t. their cost. Note that the following lemma proves that H satisfies Invariant 6.

▶ **Lemma 20.** *For $\varepsilon \leq 0.001$ we have* $\text{cost}(H) \leq 1.9412 \cdot \text{opt}(G)$.

Proof. Let $\varepsilon'|\mathcal{P}|$ be the number of degenerate paths. Note that $\text{opt}(G) \geq |\mathcal{P}|$, and therefore $\varepsilon'|\mathcal{P}| \leq \varepsilon' \text{opt}(G)$. We prove the statement by giving a factor-revealing LP, which proves $\text{cost}(H) \leq \rho \cdot \text{opt}(G)$ if the polyhedron is empty for a specific value of ρ. The parameter ρ will then give the desired approximation ratio. The factor-revealing LP is as follows. We use the same variables as before, but for simplicity we divide all constraints by $|\mathcal{P}|$. For ease of exposition, we do not replace each variable with a new variable, which is simply the old variable divided by $|\mathcal{P}|$. Instead, we simply use the same variable names.

We use the constraints we obtained by the preceding discussion and lemmas. That is, we use the constraints obtained from Lemma 16, Lemma 17, Lemma 18, Lemma 19, and constraints that we naturally obtained from the definitions of the variables. Additionally, we simplify the constraints on the optimum solution value as follows. From Lemma 16 we get $\text{opt}_C \geq 2|\mathcal{P}| - \sum_{i\geq 1} \omega_i$. Furthermore, by definition, we have $\omega_0 + \sum_{i=1}^{n} 2i \cdot \omega_i = 2$ (recall that we divide all numbers by $|\mathcal{P}|$). Since in all the constraints we used there is no other bound on γ_3, both inequalities can be simplified to $\text{opt}_C \geq 2 - \omega_1 - \omega_2 - \omega_3$ and $\omega_0 + 2\omega_1 + 4\omega_2 + 6\omega_3 \leq 2$. We then get the following polyhedron.

$$
\begin{aligned}
\text{opt}_C &\geq 2 - \omega_1 - \omega_2 - \omega_3 \\
\text{cost}(H) &\geq \rho \cdot \text{opt}_C \\
\text{cost}(H) &\leq 3 - \frac{5}{4}\alpha_1 - \alpha_2 + \frac{1}{4}\alpha_0^e + \frac{1}{2}\varepsilon' \\
\text{cost}(H) &\leq 3 - \frac{5}{4}\beta_1 - \beta_2 + \frac{1}{4}\beta_0^e + \frac{1}{2}\varepsilon' \\
\text{cost}(H) &\leq 3 - \frac{5}{4}\gamma_1 - \gamma_2 + \frac{1}{4}\gamma_0^e + \frac{1}{2}\varepsilon' \\
2 &\geq \omega_0 + 2\omega_1 + 4\omega_2 + 6\omega_3 \\
\alpha_1 &\geq \omega_1, \beta_1, \gamma_1 \\
\alpha_1 &\geq \omega_1 + \alpha_1^0 \\
\alpha_1 &\geq \omega_1 + \beta_1^0 \\
\beta_1 &\geq \omega_1 \\
\alpha_1 &\geq \alpha_0^e \\
\beta_1 &\geq \beta_0^e \\
\gamma_1 &\geq \gamma_0^e \\
\omega_0 &\geq \beta_0^e \\
\alpha_1 &= \alpha_1^0 + \alpha_1^1 + \alpha_1^2 \\
\beta_1 &= \beta_1^0 + \beta_1^1 + \beta_1^2 \\
\alpha_2 &\geq \frac{3}{5+\varepsilon} \cdot (\omega_2 - 2\alpha_1^0 - \alpha_1^1) \\
\beta_2 &\geq \frac{3}{5+\varepsilon} \cdot (\omega_2 - 2\beta_1^0 - \beta_1^1) \\
\gamma_1 + \gamma_2 &\geq \frac{3}{5+\varepsilon} \cdot (\omega_1 + \omega_2) \\
\omega_0, \omega_1, \omega_2, \omega_3 &\geq 0 \\
\alpha_0^e, \alpha_1, \alpha_1^0, \alpha_1^1, \alpha_1^2, \alpha_2 &\geq 0 \\
\beta_0^e, \beta_1, \beta_1^0, \beta_1^1, \beta_1^2, \beta_2 &\geq 0 \\
\gamma_0^e, \gamma_1, \gamma_1^0, \gamma_1^1, \gamma_1^2, \gamma_2 &\geq 0 \ .
\end{aligned}
$$

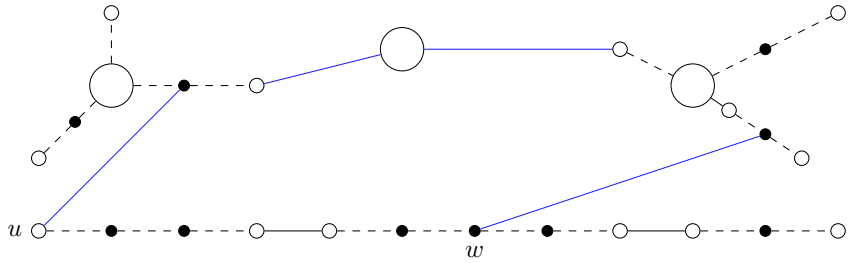


Figure 4 Example for the bridge-covering process: a pseudo-ear covering the witness path from u to w. H consists of the dashed edges (in $E(\mathcal{P})$) and solid black edges (links of H). Filled vertices are interior vertices of paths from $\mathcal{P}$, empty vertices are endvertices of paths in $\mathcal{P}$. The big circles represent 2EC blocks or components. The blue edges form a pseudo-ear Q^{uw} from u to w. All bridges, vertices and blocks on the path from u to w in H will be in a 2EC block in $H \cup Q^{uw}$.

By plugging this into a standard solver and setting $\rho \geq 1.9412$, $\varepsilon \leq 0.001$ and $\varepsilon' \leq 0.0001$, we obtain that the polyhedron is empty and hence we obtain the result. ◀

5 Bridge-Covering

After computing a good starting solution, our next goal is to ensure that the current partial solution $H = (V, E(\mathcal{P}) \cup S)$ is bridgeless. This process, called *bridge-covering*, eliminates all bridges in H while preserving Invariants 6-8. In particular, we show the following lemma.

▶ **Lemma 21.** *Let G be some structured instance of* PAP *and $H = (V, E(\mathcal{P}) \cup S)$ be a solution satisfying the Invariants 6-8. In polynomial time we can compute a solution $H' = (V, E(\mathcal{P}) \cup S')$ that satisfies Invariants 6-8 and contains no bridges.*

Throughout this section we assume that H contains some complex component C (a component that contains bridges), as otherwise the lemma holds trivially. We prove Lemma 21 by showing that we can iteratively turn H into a solution H' such that H' also satisfies the Invariants 6-8 and additionally H' contains fewer bridges than H (except for one case that we need for technical reasons). Applying this procedure exhaustively proves Lemma 21.

Intuitively, as long as our current solution H contains complex components, i.e., it contains bridges, our goal is to add certain links to H in order to "cover" the bridges and, at the same time, maintain the Invariants 6-8. To do so, it will be convenient to add so-called pseudo-ears to H, defined as follows. Similar definitions have been used in [8] and [23].

▶ **Definition 22** (Pseudo-ear and Witness Path)**.** *Let C be a complex component of H and $u, v \in V(C)$ be distinct vertices in G_C. A* pseudo-ear *Q^{uv} from u to v is a simple path in G_C that starts in u and ends in v and does not visit any vertex of C except u and v. The unique path W^{uv} from u to v in H_C is called the* witness path *of Q^{uv}. Note that the edges of W^{uv} are bridges of C.*

An example for a pseudo-ear is given in Figure 4. We first show that a pseudo-ear Q^{uv} can be found in polynomial time, if one exists. Furthermore, pseudo-ears are useful in the following sense. If, for some pseudo-ear Q^{uv}, its witness path W^{uv} contains "enough" credit, e.g., if W^{uv} contains 2 block-nodes, 2 lonely vertices, or a lonely vertex or block node and 2 bridges from $H \cap L$, then we can add Q^{uv} to H to obtain the graph H'. Afterward, H' is a graph that contains fewer bridges than H, the number of connected components in H' is at most the number of connected components in H, and H' satisfies all invariants.

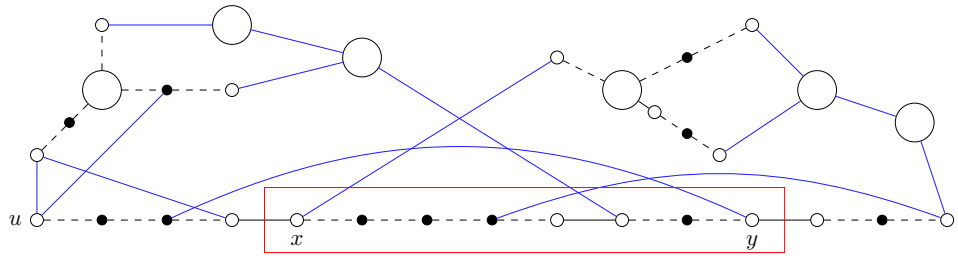


Figure 5 No witness path of a pseudo-ear starting in u contains 2 bridges of $H \cap L$. Note that the two paths P_1 and P_2 containing x and y (vertices in red box), respectively, form a separator in G (in fact, a P^2-separator).

Using pseudo-ears, we then show that in polynomial time we can find link sets $R \subseteq L \setminus S$ and $F \subseteq S$, such that $(H \cup R) \setminus F$ has fewer bridges than H and also satisfies all invariants. This then proves Lemma 21. It is crucial for this step that G is structured: We usually argue that G must have a desired pseudo-ear, as otherwise the absence of it implies that G is not structured, a contradiction.

We give an intuition for how we use structured graphs to prove Lemma 21 (see also Figure 5): If from a lonely leaf vertex u (or a leaf block node) there exists a pseudo-ear such that its witness path contains at least one other leaf vertex, one block node, or 2 bridges in L, then we can add this pseudo-ear to H, and we reduced the number of bridges while satisfying all invariants, as desired. Otherwise, if there exists no such pseudo-ear, then this complex component C must look locally like a path. Furthermore, the absence of this pseudo-ear implies that in C there must be two paths $P_1, P_2 \in \mathcal{P}$ such that $V(P_1) \cup V(P_2)$ is a separator in G. However, since C locally looks like a path, there is a link e in C connecting one endvertex of P_1 with one endvertex of P_2 in C. But then P_1 and P_2 together with e form a P^2-separator, which is a contradiction to G being structured (Property (P4)). Thus, there must be a desired pseudo-ear, which we can find in polynomial time, and hence we make progress. This is the whole intuition for proving Lemma 21. However, there are many corner cases that we have to deal with, which makes the proof a bit lengthy.

6 Gluing Algorithm

After the bridge-covering step, the graph $H = (V, P \cup S)$ consists of multiple 2-edge-connected components that remain disconnected from each other. The final step of our algorithm, called *gluing*, ensures that these components are merged into a single 2-edge-connected graph while maintaining the approximation guarantee. We show the following.

▶ **Lemma 23.** *Let G be some structured instance of* PAP *and $H = (V, E(\mathcal{P}) \cup S)$ be a solution satisfying the Invariants 6-8 that contains no bridges. In polynomial time we can compute a solution $H' = (V, E(\mathcal{P}) \cup S')$ satisfying the Invariants 6-8 such that H' is a 2-edge-connected spanning subgraph.*

Throughout this section we are given a solution S such that $H = (V, E(\mathcal{P}) \cup S)$ consists of several 2-edge-connected components, and each component contains at least the vertex set of two distinct paths $P, P' \in \mathcal{P}$. According to the credit scheme (Part (E)), each component has a credit of 2 if it is large, or it is small and contains a degenerate path. Otherwise, if it is small and not degenerate, it has a credit of $\frac{3}{2}$. Recall that a 2EC component is small if it contains exactly 2 links.

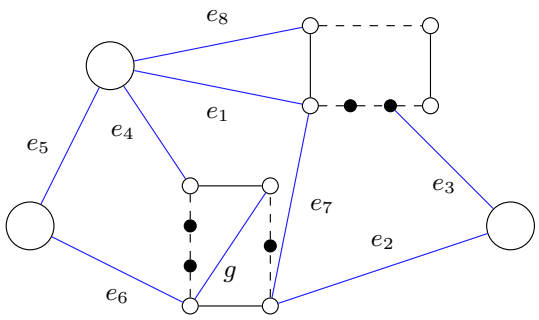


Figure 6 Example for the gluing algorithm and a good cycle. The bridgeless 2-edge-cover consists of large components (big circles) and small components (short cycles). H consists of the dashed edges (in $E(\mathcal{P})$) and solid black edges (links of H). Blue edges are edges in $E(G) \setminus E(H)$. The cycle $e_4e_5e_6$ is a good cycle in the component graph as it contains 2 large components. The cycle $e_4e_1e_7$ is also a good cycle as it shortcuts a small component using the Hamiltonian path through g.

Our approach here is to find *good* cycles in the component graph $G^{\text{comp}}(H)$, that is, the graph arising from G by contracting each 2-edge-connected component of H to a single vertex. Adding cycles from the component graph to H reduces the number of components and does not introduce any bridges. By iteratively adding such cycles, we eventually end up with a single 2-edge-connected component, and we are done.

Good cycles are cycles that either contain 2 components with credit 2 each, or a small component that can be *shortcut*. A small component C can be shortcut w.r.t. a cycle K of $G^{\text{comp}}(H)$ if we can replace $E(C)$ by a Hamiltonian path $Q \subseteq G[V(C)]$ such that $|Q \cap L| < |E(C) \cap L|$ and the vertices of C that are incident to K are the endvertices of Q. If a good cycle contains a shortcut, we then update H to $H' = (H \setminus E(C)) \cup K \cup Q$ and refer to this operation as *augmenting* H with K. Two examples of good cycles can be found in Figure 6. We then show that 1) augmenting H with a good cycle results in a graph with fewer components that is bridgeless and satisfies all invariants, 2) we can find good cycles in polynomial time if one exists, and 3) the component graph of a structured graph always admits a good cycle. This then proves Lemma 23. To prove 3), we use that G is structured: We show that every small component that does not contain a degenerate path is incident to 2 edges in G that can be used to construct a shortcut. Furthermore, since G is structured and contains no C^2-separator, these 2 edges are contained in some cycle in the component graph $G^{\text{comp}}(H)$, which we can find in polynomial time.

7 Proof of Theorem 1

Theorem 1 now easily follows from Lemmas 5-23.

Proof of Theorem 1. For structured graphs G, we first use Lemma 9 to compute a solution S such that $H = (V, E(\mathcal{P}) \cup S)$ satisfies Invariants 6-8. Lemma 21 now guarantees that we can transform H into a solution H' that is bridgeless and satisfies Invariants 6-8. Finally, Lemma 23 ensures that we can obtain in polynomial time a 2-edge-connected spanning subgraph $H'' = (V, E(\mathcal{P}) \cup S'')$ satisfying Invariants 6-8. In particular, H'' is a feasible solution and satisfies $\text{cost}(H'') = |S''| + \text{credits}(H'') \leq 1.9412 \cdot \text{opt}$. Since $\text{credits}(H'') \geq 0$, we have $|S''| \leq 1.9412 \cdot \text{opt}$, implying that our algorithm is a polynomial-time 1.9412-approximation for structured graphs. Finally, Lemma 5 then guarantees that this leads to a $(1.9412 + \varepsilon)$-approximation for arbitrary instances of bounded-length PAP. However,

we mention here that the guarantee of 1.9412 from Lemma 9 has some slack, and hence we can choose ε small enough (still constant) so that we have a 1.9412-approximation for bounded-length PAP. ◀

8 Conclusion and Future Work

In this paper we designed a 1.9412-approximation for the Path Augmentation problem, where the length of each path is bounded. We established several key ingredients: (1) a $(\frac{11}{6}+\varepsilon)$-approximation preserving reduction to structured instances of bounded-length PAP; (2) a new relaxation of the problem together with a related packing problem, which we used to obtain a good starting solution using a factor-revealing LP; (3) leveraging concepts of other connectivity augmentation problems to turn the starting solution into a feasible solution.

There might be room for improvement, in particular in bounding the cost of the starting solution. Obtaining a better bound on the cost of the starting solution, while still maintaining the credit invariants, directly gives an improved approximation ratio for bounded-length PAP. However, to improve the approximation ratio below $\frac{11}{6}$ requires new insights.

We emphasize that the requirement of bounded-length instances of PAP is only needed for the reduction to structured graphs; the remaining parts of the algorithm also work for unbounded-length PAP as long as the input is structured. Hence, it would be nice to see if this assumption can be dropped without any loss in the approximation guarantee.

References

1 David Adjiashvili. Beating approximation factor two for weighted tree augmentation with bounded costs. *ACM Transactions on Algorithms (TALG)*, 15(2):1–26, 2018. doi:10.1145/3182395.

2 Étienne Bamas, Marina Drygala, and Ola Svensson. A simple lp-based approximation algorithm for the matching augmentation problem. In Karen I. Aardal and Laura Sanità, editors, *Integer Programming and Combinatorial Optimization - 23rd International Conference, IPCO 2022, Eindhoven, The Netherlands, June 27-29, 2022, Proceedings*, volume 13265 of *Lecture Notes in Computer Science*, pages 57–69. Springer, 2022. doi:10.1007/978-3-031-06901-7_5.

3 Miguel Bosch-Calvo, Mohit Garg, Fabrizio Grandoni, Felix Hommelsheim, Afrouz Jabal Ameli, and Alexander Lindermayr. A 5/4-approximation for two-edge connectivity. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 653–664, 2025. doi:10.1145/3717823.3718275.

4 Miguel Bosch-Calvo, Fabrizio Grandoni, and Afrouz Jabal Ameli. A 4/3 approximation for 2-vertex-connectivity. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPIcs*, pages 29:1–29:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ICALP.2023.29.

5 Jaroslaw Byrka, Fabrizio Grandoni, and Afrouz Jabal Ameli. Breaching the 2-approximation barrier for connectivity augmentation: A reduction to steiner tree. *SIAM J. Comput.*, 52(3):718–739, 2023. doi:10.1137/21M1421143.

6 Federica Cecchetto, Vera Traub, and Rico Zenklusen. Bridging the gap between tree and connectivity augmentation: unified and stronger approaches. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 370–383, 2021. doi:10.1145/3406325.3451086.

7 Joe Cheriyan, Jack Dippel, Fabrizio Grandoni, Arindam Khan, and Vishnu V Narayan. The matching augmentation problem: a 7 4-approximation algorithm. *Mathematical Programming*, 182(1-2):315–354, 2020. doi:10.1007/s10107-019-01394-z.

8 Joseph Cheriyan, Robert Cummings, Jack Dippel, and Jasper Zhu. An improved approximation algorithm for the matching augmentation problem. *SIAM Journal on Discrete Mathematics*, 37(1):163–190, 2023. doi:10.1137/21M1453505.

9 Joseph Cheriyan and Zhihan Gao. Approximating (unweighted) tree augmentation via lift-and-project, part i: stemless tap. *Algorithmica*, 80:530–559, 2018. doi:10.1007/s00453-016-0270-4.

10 Joseph Cheriyan and Zhihan Gao. Approximating (unweighted) tree augmentation via lift-and-project, part ii. *Algorithmica*, 80:608–651, 2018. doi:10.1007/s00453-017-0275-7.

11 Joseph Cheriyan, Howard Karloff, Rohit Khandekar, and Jochen Könemann. On the integrality ratio for tree augmentation. *Operations Research Letters*, 36(4):399–401, 2008. doi:10.1016/j.orl.2008.01.009.

12 Joseph Cheriyan, András Sebo, and Zoltán Szigeti. Improving on the 1.5-approximation of a smallest 2-edge connected spanning subgraph. *SIAM Journal on Discrete Mathematics*, 14(2):170–180, 2001. doi:10.1137/S0895480199362071.

13 Joseph Cheriyan and Ramakrishna Thurimella. Approximating minimum-size k-connected spanning subgraphs via matching. *SIAM J. Comput.*, 30(2):528–560, 2000. doi:10.1137/S009753979833920X.

14 Nachshon Cohen and Zeev Nutov. A (1+ ln2)-approximation algorithm for minimum-cost 2-edge-connectivity augmentation of trees with constant radius. *Theoretical Computer Science*, 489:67–74, 2013. doi:10.1016/j.tcs.2013.04.004.

15 Marek Cygan. Improved approximation for 3-dimensional matching via bounded pathwidth local search. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 509–518. IEEE, 2013. doi:10.1109/FOCS.2013.61.

16 Artur Czumaj and Andrzej Lingas. On approximability of the minimum-cost k-connected spanning subgraph problem. In Robert Endre Tarjan and Tandy J. Warnow, editors, *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, 17-19 January 1999, Baltimore, Maryland, USA*, pages 281–290. ACM/SIAM, 1999. doi:10.5555/314500.314573.

17 Guy Even, Jon Feldman, Guy Kortsarz, and Zeev Nutov. A 1.8 approximation algorithm for augmenting edge-connectivity of a graph from 1 to 2. *ACM Transactions on Algorithms (TALG)*, 5(2):1–17, 2009. doi:10.1145/1497290.1497297.

18 Cristina G. Fernandes. A better approximation ratio for the minimum size k-edge-connected spanning subgraph problem. *J. Algorithms*, 28(1):105–124, 1998. doi:10.1006/jagm.1998.0931.

19 Samuel Fiorini, Martin Groß, Jochen Könemann, and Laura Sanità. Approximating weighted tree augmentation via chvátal-gomory cuts. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 817–831. SIAM, 2018. doi:10.1137/1.9781611975031.53.

20 Greg N Frederickson and Joseph Ja'Ja'. Approximation algorithms for several graph augmentation problems. *SIAM Journal on Computing*, 10(2):270–283, 1981. doi:10.1137/0210019.

21 Martin Fürer and Huiwen Yu. Approximating the-set packing problem by local improvements. In *International Symposium on Combinatorial Optimization*, pages 408–420. Springer, 2014. doi:10.1007/978-3-319-09174-7_35.

22 Mohit Garg, Fabrizio Grandoni, and Afrouz Jabal Ameli. Improved approximation for two-edge-connectivity. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2368–2410. SIAM, 2023. doi:10.1137/1.9781611977554.ch92.

23 Mohit Garg, Felix Hommelsheim, and Nicole Megow. Matching augmentation via simultaneous contractions. In *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, pages 65:1–65:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ICALP.2023.65.

24 Naveen Garg, Santosh S. Vempala, and Aman Singla. Improved approximation algorithms for biconnected subgraphs via better lower bounding techniques. In Vijaya Ramachandran, editor, *Proceedings of the Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 25-27 January 1993, Austin, Texas, USA*, pages 103–111. ACM/SIAM, 1993. doi:10.5555/313559.313618.

25 Fabrizio Grandoni, Afrouz Jabal Ameli, and Vera Traub. Breaching the 2-approximation barrier for the forest augmentation problem. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1598–1611, 2022. doi:10.1145/3519935.3520035.

26 Fabrizio Grandoni, Christos Kalaitzis, and Rico Zenklusen. Improved approximation for tree augmentation: saving by rewiring. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 632–645, 2018. doi:10.1145/3188745.3188898.

27 Klaus Heeger and Jens Vygen. Two-connected spanning subgraphs with at most $\frac{10}{7}OPT$ edges. *SIAM J. Discret. Math.*, 31(3):1820–1835, 2017. doi:10.1137/16M1091587.

28 Felix Hommelsheim. Improved approximation algorithms for path and forest augmentation via a novel relaxation. *CoRR*, abs/2505.15324, 2025. doi:10.48550/arXiv.2505.15324.

29 Felix Hommelsheim, Alexander Lindermayr, and Zhenwei Liu. A better-than-5/4-approximation for two-edge connectivity. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1468–1520. SIAM, 2026. doi:10.1137/1.9781611978971.54.

30 Christoph Hunkenschröder, Santosh Vempala, and Adrian Vetta. A 4/3-approximation algorithm for the minimum 2-edge connected subgraph problem. *ACM Transactions on Algorithms (TALG)*, 15(4):1–28, 2019. doi:10.1145/3341599.

31 Kamal Jain. A factor 2 approximation algorithm for the generalized steiner network problem. *Combinatorica*, 21:39–60, 2001. doi:10.1007/s004930170004.

32 Samir Khuller and Ramakrishna Thurimella. Approximation algorithms for graph augmentation. *Journal of algorithms*, 14(2):214–225, 1993. doi:10.1006/jagm.1993.1010.

33 Samir Khuller and Uzi Vishkin. Biconnectivity approximations and graph carvings. *Journal of the ACM (JACM)*, 41(2):214–235, 1994. doi:10.1145/174652.174654.

34 David G Kirkpatrick and Pavol Hell. On the completeness of a generalized matching problem. In *Proceedings of the tenth annual ACM symposium on Theory of computing*, pages 240–245, 1978. doi:10.1145/800133.804353.

35 Yusuke Kobayashi and Takashi Noguchi. An approximation algorithm for two-edge-connected subgraph problem via triangle-free two-edge-cover. *arXiv preprint arXiv:2304.13228*, 2023. doi:10.4230/LIPIcs.ISAAC.2023.49.

36 Guy Kortsarz and Zeev Nutov. A simplified 1.5-approximation algorithm for augmenting edge-connectivity of a graph from 1 to 2. *ACM Transactions on Algorithms (TALG)*, 12(2):1–20, 2015. doi:10.1145/2786981.

37 Guy Kortsarz and Zeev Nutov. Lp-relaxations for tree augmentation. *Discrete Applied Mathematics*, 239:94–105, 2018. doi:10.1016/j.dam.2017.12.033.

38 Hiroshi Nagamochi. An approximation for finding a smallest 2-edge-connected subgraph containing a specified spanning tree. *Discrete Applied Mathematics*, 126(1):83–113, 2003. doi:10.1016/S0166-218X(02)00218-4.

39 Zeev Nutov. On the tree augmentation problem. *Algorithmica*, 83:553–575, 2021. doi:10.1007/s00453-020-00765-9.

40 Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*, volume 24 of *Algorithms and Combinatorics*. Springer, Berlin, Heidelberg, 2003. URL: https://link.springer.com/book/9783540443896.

41 András Sebő and Jens Vygen. Shorter tours by nicer ears: 7/5-approximation for the graph-tsp, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. *Combinatorica*, pages 1–34, 2014. doi:10.1007/s00493-014-2960-3.

42 Vera Traub and Rico Zenklusen. Better-than-2 approximations for weighted tree augmentation and applications to steiner tree. *J. ACM*, 72(2):16:1–16:40, 2025. doi:10.1145/3722101.

Capacitated Partition Vertex Cover and Partition Edge Cover

Rajni Dabas
Northwestern University, Evanston, IL, USA

Samir Khuller
Northwestern University, Evanston, IL, USA

Emilie Rivkin
Northwestern University, Evanston, IL, USA

Abstract

We study generalizations of the classical Vertex Cover and Edge Cover problems that incorporate group-wise coverage.

Our first focus is the Capacitated Partition Vertex Cover (C-PVC) problem in hypergraphs. In C-PVC, we are given a hypergraph with capacities on its vertices and a partition of the hyperedge set into ω distinct groups. The objective is to select a minimum size subset of vertices that satisfies two main conditions: (1) in each group, the total number of covered hyperedges meets a specified threshold, and (2) the number of hyperedges assigned to any vertex respects its capacity constraint. A covered hyperedge is required to be assigned to a selected vertex that belongs to the hyperedge. This formulation generalizes classical Vertex Cover, Partial Vertex Cover, and Partition Vertex Cover.

We investigate two primary variants: soft capacitated (multiple copies of a vertex are allowed) and hard capacitated (each vertex can be chosen at most once). Let f denote the rank of the hypergraph (i.e., the maximum number of vertices contained in any single hyperedge). Our main contributions are: (i) an $(f+1)$-approximation algorithm for the weighted soft-capacitated C-PVC problem, which runs in polynomial time for constant ω, and (ii) an $(f+\epsilon)$-approximation algorithm for the unweighted hard-capacitated C-PVC problem, which runs in $n^{O(\omega/\epsilon)}$ time.

We also study a natural generalization of the edge cover problem, the *Weighted Partition Edge Cover* (W-PEC) problem, where each edge has an associated weight, and the vertex set is partitioned into groups. For each group, the goal is to cover at least a specified number of vertices using incident edges, while minimizing the total weight of the selected edges. We present the first exact polynomial-time algorithm for the weighted case, improving runtime from $O(\omega n^3)$ to $O(mn + n^2 \log n)$ and simplifying the algorithmic structure over prior unweighted approaches (that rely on the tropical matching problem).

2012 ACM Subject Classification Theory of computation → Packing and covering problems

Keywords and phrases Approximation algorithms, capacitated vertex cover, iterative rounding

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.8

Category APPROX

Funding Supported by NSF 2216970 (IDEAL) as well as a gift from Adobe Labs.

Acknowledgements We would like to thank Amol Deshpande, Hal Gabow, Tanmay Inamdar, Julian Mestre and Emily Pitler for useful discussions.

1 Introduction

The classical *Vertex Cover* (VC) problem in graphs is defined as follows: given a graph $G = (V, E)$ with weights on the vertices, find a minimum size subset of vertices S so that every edge of the graph is covered, in other words, at least one of its end points is in S. This


© Rajni Dabas, Samir Khuller, and Emilie Rivkin;
licensed under Creative Commons License CC-BY 4.0

extremely simple problem has a rich history. It was one of the earliest problems shown to be NP-complete by Karp [30] and was shown to have a polynomial time solution when the graph is bipartite. On the hardness side, [31] assumes the Unique Games Conjecture and shows that Vertex Cover is hard to approximate within a factor $(2-\epsilon)$ for every $\epsilon > 0$. Gavril [22] showed that a simple algorithm that takes all the vertices of any maximal matching is a 2-approximation (because the cardinality of any maximal matching is a lower bound on the optimal solution size). Subsequently, this simple algorithm was extended to weighted[1] graphs as well, but this required significantly deeper understanding of the role played by linear programming (LP) relaxations [24] and finally a simple combinatorial algorithm was developed based on LP duality [6]. This problem also paved the way for improvements and the well known local ratio technique [7] that found numerous applications. In addition, approximation algorithms easily follow from the work of Nemhauser and Trotter who showed that an optimal LP solution has a very simple structure [35]. In the Fixed Parameter Tractability arena, again Vertex Cover has played a crucial role in the understanding of kernelization and efficient algorithms.

Several extensions of VC have been studied. One major direction introduces capacities on vertices, leading to the *Capacitated Vertex Cover* (CVC) problem. Here, each selected vertex may cover only a limited number of incident edges. In this problem, a key distinction arises between *soft capacities* (where multiple copies of a vertex can be purchased) and *hard capacities* (where each vertex is chosen at most once). For soft capacities, Guha et al.[23] first introduced the problem and provided a 2-approximation using a primal-dual algorithm; another 2-approximation using dependent randomized rounding was later given by Gandhi et al. [20]. Notably, these results often hold for the weighted setting where vertices have costs. The hard capacity variant (VC-HC) presents significantly more challenges. Chuzhoy and Naor [12] initiated its study, giving a 3-approximation for the unweighted version on simple graphs. Crucially, they also showed that the weighted version of VC-HC is at least as hard as Set Cover, which is NP-hard to approximate within a factor of $(1-\epsilon)\ln n$ for every $\epsilon > 0$ [16]. Focusing on the unweighted VC-HC, Gandhi et al. [21] improved the approximation to a tight 2. When generalizing these problems to hypergraphs, approximation guarantees typically depend on the maximum size of a hyperedge. Let f denote the rank of the hypergraph, defined as the maximum number of vertices contained in any single hyperedge. A hypergraph whose rank is bounded by f is commonly referred to as an f-hypergraph. For f-hypergraphs, Saha and Khuller [36] provided the first $O(f)$-approximation (specifically $\min\{6f, 65\}$). Cheung, Goemans, and Wong [11] then improved the bounds to $2f$ using a deterministic two-stage LP rounding approach. The gap was finally closed to a tight f-approximation for unweighted VC-HC on f-hypergraphs, achieved independently by Kao [28] and Wong [37], using iterative rounding techniques.

A separate parallel line of work relaxes the requirement to cover all edges. *Partial Vertex Cover* asks to cover at least a specified number of edges [10, 5, 21]. A further generalization is *Partition Vertex Cover* (PVC), where the edge set is partitioned into ω groups, the solution must cover a specified minimum number of edges from each group. PVC was introduced by Bera et al. [9], who obtained an $O(\log \omega)$-approximation via a strengthened LP with exponentially many constraints. They also proved that the problem is at least as hard as Set Cover when ω is unbounded. For f-hypergraphs, Hong and Kao [25] obtained a $(f+1)\log \omega$ approximation for the uncapacitated PVC problem. This is particularly relevant as the techniques in Bera et al. for graphs do not extend naturally to hypergraphs.

[1] Every vertex has a weight and the aim is to find a minimum weighted subset of vertices.

For constant ω, and unweighted graphs, Bandyapadhyay et al. [3] gave a $(2+\epsilon)$-approximation in time $n^{O(\omega/\epsilon)}$, later improved to the weighted case with running time $n^{O(\omega)}$ independently by Liu and Li [32] and Dabas et al. [13].

Combining both dimensions - vertex capacities and partition-based coverage - yields richer and significantly more challenging models. While Partial VC with capacities has been investigated[2], these methods fail when coverage requirements are partitioned across multiple groups. In this paper, we close this gap by introducing the *Capacitated Partition Vertex Cover* (C-PVC) problem, which unifies the capacitated and partition variants of the Vertex Cover problem. We study the problem in the general setting of hypergraphs.

Beyond theoretical generalization, C-PVC provides a framework for fair resource allocation in systems where demands can be served by multiple potential providers. For example, companies like Amazon solve a massive "Capacitated Coverage" problem to assign specific fulfillment centers to zip codes for same-day delivery. In 2016, Bloomberg published an investigation showing that Amazon's Prime Free Same-Day Delivery excluded predominantly Black neighborhoods (like Roxbury in Boston and the Bronx in New York) while covering the white neighborhoods surrounding them [26]. Similar fairness concerns arise in urban mobility, where car-sharing fleets are often concentrated in high-income and advantaged zones, failing to improve accessibility in underserved communities [15]. The direct applicability of the C-PVC framework stems from the fact that these resource allocation challenges are naturally modeled as Hypergraph Vertex Cover problems rather than simple graph problems. C-PVC formulation addresses equity by enforcing a coverage threshold ρ_g for each group ensuring that every demographic/geographic sector receives a minimum standard of service.

▶ **Definition 1** (Capacitated Partition Vertex Cover (C-PVC)). *We are given a hypergraph $H=(V,E)$, where $f=\max_{e\in E}|e|$ is the rank of the hypergraph. We are also given, a vertex weight function $w: V\to\mathbb{R}_{\geq 0}$, a vertex capacity function $k: V\to\mathbb{Z}^+$, a partition of the hyperedge set $E=E_1\cup E_2\cup\cdots\cup E_\omega$, and a coverage requirement (threshold) $\rho_g\in\mathbb{Z}^+$ for each group E_g.*

We denote the capacity of a vertex v by k_v and its weight by w_v. A feasible solution consists of a set of integer copy counts $\{x_v\}_{v\in V}$ (where $x_v\in\mathbb{Z}_{\geq 0}$) and an assignment $\varphi: E'\to V$ for a subset of covered edges $E'\subseteq E$, satisfying:

1. ***Valid Assignment:*** *For every edge $e\in E'$, the assigned vertex must be in the edge, i.e., $\varphi(e)\in e$.*
2. ***Capacity Constraint:*** *The number of edges assigned to any vertex v is at most its total capacity:*

$$|\{e\in E' \mid \varphi(e)=v\}|\leq x_v\cdot k_v \quad \forall v\in V$$

3. ***Partition Constraint:*** *For each group $g\in[\omega]$, the number of covered edges from that group meets its threshold:*

$$|E'\cap E_g|\geq \rho_g$$

The objective is to find a feasible solution that minimizes $\sum_{v\in V} x_v\cdot w_v$. We refer to this as the soft-capacitated variant. The hard-capacitated variant adds the constraint that $x_v\in\{0,1\}$ for all $v\in V$.

[2] For hard capacities, Kao et al [29] give an f-approximation using iterative rounding. For soft capacities, Bar-Yehuda et al [8] give an f-approximation using the local ratio method and Mestre [34] gives a 2-approximation using a primal-dual algorithm.

In this work, we present two algorithms for the C-PVC problem. We obtain an $(f+1)$-approximation for the soft-capacitated weighted case.(Theorem 2) and an $(f+\epsilon)$-approximation for the hard-capacitated unweighted case (Theorem 3). Both algorithms run in polynomial time with respect to n and m but exhibit exponential dependence on ω, which yields an efficient polynomial-time solution when the number of groups ω is constant. This dependence on ω is consistent with previous work on partition-constrained problems. While the exact problem is known to be $W[1]$-hard parameterized by the number of groups ω (which rules out exact FPT algorithms), the existence of an FPT approximation algorithm parameterized by ω remains an intriguing open question.

For the weighted variant of the hard-capacitated case, Chuzhoy and Naor [12] showed a hardness of $\Omega(\log n)$ of any poly-time approximation. Briefly, we can give the same upper bound for Hard C-PVC. Let H be a hypergraph of rank f. For each group $g \in [\omega]$, add one vertex v_g with capacity $|E_g| - \rho_g$ (the allowed number of uncovered edges of that group) and $w_{v_g} = 0$. For each hyperedge $e \in E_g$, add v_g to e. Now, we can solve a full-coverage instance of weighted vertex cover with hard capacities on hypergraph H' with rank $f+1$ using Chuzhoy and Naor's approximation algorithm for hypergraph vertex cover. This[3] will give an approximation with factor $O(\log(n+\omega)) = O(\log(n))$ for constant ω. For this reason, we only consider the unweighted variant of Hard C-PVC.

▶ **Theorem 2.** *There exists a $(f+1)$-factor approximation algorithm for the* Soft Capacitated Weighted Partition Vertex Cover *problem that runs in polynomial time for constant ω.*

▶ **Theorem 3.** *For any fixed $\epsilon > 0$, there exists an $(f+\epsilon)$-factor approximation algorithm for the* Hard Capacitated Unweighted Partition Vertex Cover *problem that runs in time $n^{O(\omega/\epsilon)}$.*

While the vertex cover problem focuses on selecting a subset of vertices to cover all edges, a natural dual problem is the *edge cover* problem. In this variant, the goal is to select a subset of edges such that every vertex is incident to at least one selected edge. Although structurally similar to vertex cover, the edge cover problem has very different algorithmic properties—it can be solved in polynomial time via reductions to maximum matching—and it arises in applications such as network design and resource allocation.

In this paper, we study a natural generalization of the edge cover problem called the *Partition Edge Cover* (Partition-EC) problem which is defined as follows.

▶ **Definition 4** (Weighted Partition-EC)**.** *We are given a graph $G = (V, E)$ with a weight function $p : E \to \mathbb{R}_{\geq 0}$, a partition of the vertex set $V = V_1 \cup \cdots \cup V_\omega$, and a parameter ρ_g for each group. We want to find a minimum-weight subset of edges such that at least ρ_g vertices from group V_g are covered.*

Bandyapadhyay et al. [4] studied this problem in the unweighted setting and gave an exact polynomial-time algorithm with a runtime of $O(\omega n^3)$. Their approach first reduces the problem to *Budgeted Matching*, which is then reduced to *Tropical Matching*, allowing the use of known algorithms for the latter to obtain an optimal solution.

In this work, we present a simpler algorithm with improved running time (Theorem 5). Our algorithm avoids the use of tropical matching and runs in time $O(mn + n^2 \log n)$. Moreover, our approach naturally extends to the more general *weighted* setting, where edges have arbitrary non-negative weights.

[3] This does not generalize to soft capacities, since we could pick unlimited copies of these additional zero weight vertices.

▶ **Theorem 5.** *The Weighted Partition Edge Cover problem can be solved exactly in* $O(mn + n^2 \log n)$ *time.*

Related Work. The closely related k-center problem has also been well studied in the partition framework. The first algorithm was proposed by Bandyapadhyay et al. [4], providing a polynomial time 2-approximation while allowing $k+\omega$ centers. A subsequent result by Anegg et al. [1] eliminated the violation in cardinality of centers and obtained a 4-approximation in time $O(n^\omega)$, which was later improved to a 3-approximation in time $O(n^{\omega^2})$ by Jia et al. [27]. The same framework for facility location and k-median objectives has also been studied; see, for example, Dabas et al. [14], Bajpai et. al. [2], and the references therein.

Organization of the paper. The remainder of the paper is organized as follows. We highlight the technical challenges and key ideas used to achieve our results in Section 1.1. Sections 2 and 3 present the $(f+1)$-approximation and $(f+\epsilon)$-approximation algorithms for soft and hard capacitated PVC, respectively. In Section 4, we describe the algorithm for W-PEC. Proofs marked by (†) are deferred to the Appendix.

1.1 Technical Challenges and Key Ideas

Covering problems are extensively studied, but partial versions of these questions (that require coverage of a given number of elements) make the problems much harder and significant new ideas and techniques were required to obtain approximation algorithms for Facility Location, k-median, k-center, Vertex Cover, etc. Despite the seemingly "small" generalization from $\omega = 1$ to $\omega > 1$, the problems again become much harder. For example, for the k-center problem with outliers, there is a simple greedy algorithm with a factor 3 and a simple LP rounding algorithm with a factor 2. As mentioned in the related work, the problem for $\omega > 1$ is considerably more involved; the best approximation factor of 3 requires a very high run time of $O(n^{\omega^2})$.

The same situation also arises for Vertex Cover, where for $\omega = 1$ there is basically no fundamental difference between full coverage and partial coverage and a matching bound of 2 can be obtained. But for $\omega > 1$ there is still no combinatorial algorithm that gives a $O(1)$ bound.

Soft C-PVC. For the soft-capacitated variant, the main challenge is managing fractional values in the LP relaxation while respecting the lower bound threshold for each group. Our algorithm first transforms an optimal fractional solution so that edge coverage is handled either by an already rounded high vertex or by a unique low vertex. We then formulate a configuration LP to handle low vertices. A configuration for a low vertex records, for each group g, how many edges of E_g are served by one copy of that vertex (therefore has one block of variables for each low vertex) and ω coverage constraints. The key insight here is that this configuration LP has at most ω fractional vertex blocks. We round one copy for each fractional block and retain its fractional configuration mixture - a max-flow computation determines the actual edge assignments. In the weighted case, we control the additive cost of the at most ω rounded blocks by guessing the ω most expensive relevant vertices. Our approach also eliminates the enumerative step for the unweighted case, thereby removing the running time's dependence on ϵ present in Bandyapadhyay et al. [3]. While this $(f+1)$ ratio is higher than the graph-specific $(2+\epsilon)$ ratio in prior work, the removal of $1/\epsilon$ from the exponent of the runtime is a significant algorithmic simplification and efficiency gain, particularly for the weighted case.

Hard C-PVC. Our algorithm uses an iterative rounding algorithm to fix vertices and assignments with large fractional values in each iteration. A final enumeration handles the remaining small solution space. This structure matches the known tight techniques for hard-capacitated hypergraph VC [37] but the core technical difficulty stems from integrating the ω partition constraints into the iterative rounding framework. The algorithm in Wong [37] crucially rely on a *counting argument* applied to the final LP's extreme point solution. This argument bounds the cost of rounding the remaining "small" fractional vertices by bounding their total number. However, introducing ω new partition constraints fundamentally alters the structure of this extreme point, as these ω constraints can also be tight. This breaks the standard counting argument. Our key technical contribution is a modified iterative algorithm that demonstrates that the number of remaining fractional variables is bounded by a function of ω. We then manage this remaining set via a final enumeration step, yielding an $(f + \epsilon)$-approximation algorithm whose runtime depends exponentially on ω.

Weighted Partition-EC. Prior work by Bandyapadhyay et al. [4] for the unweighted setting relied on a reduction to *Tropical Matching*. A major challenge is that it is not clear whether the tropical matching-based approach can be extended to handle weighted instances, and the existing method requires heavy algorithmic machinery. We circumvent this by identifying a simpler structure that allows us to avoid tropical matching entirely. By reducing the problem to a simpler matching formulation, we are able to handle arbitrary weights and significantly improve the runtime to $O(mn + n^2 \log n)$.

2 Soft-Capacitated Weighted Partition Vertex Cover

We first present an integer program for the SC-PVC problem. Note that we add an upper bound of m on the multiplicity x_v, since any vertex v has $k_v \geq 1$ and $|\delta(v)| \leq m$.

$$\begin{aligned}
\text{minimize} \quad & \sum_{v \in V} w_v x_v && && \text{(IP-SC)}\\
\text{subject to} \quad & \sum_{v \in e} y_{e,v} \leq 1 && \forall e \in E\\
& \sum_{e \in E_g} \left(\sum_{v \in e} y_{e,v} \right) \geq \rho_g && \forall g \in \{1, \ldots, \omega\}\\
& \sum_{e \in \delta(v)} y_{e,v} \leq k_v x_v && \forall v \in V\\
& 0 \leq y_{e,v} \leq x_v && \forall v \in e \in E\\
& x_v \in \{0, 1, \ldots, m\} && \forall v \in V\\
& y_{e,v} \in \{0, 1\} && \forall v \in e \in E
\end{aligned}$$

We will say that an integral solution is *feasible* if we have an integral multiplicity x_v for each vertex and an assignment function $\varphi : E \to V \cup \{\perp\}$ such that $|\{e \in E_g : \varphi(e) \neq \perp\}| \geq \rho_g$ for all $g \in [\omega]$ and $|\{e \in \delta(v) : \varphi(e) = v\}| \leq k_v x_v$.

We only need x to be integral in a feasible solution $\langle x, y \rangle$ in order to find a fully integral solution. The following lemma is utilized in several prior works on capacitated vertex cover and involves a simple reduction to a network flow problem.

▶ **Lemma 6** (†). *Let $\langle x, y \rangle$ be a feasible solution to the linear relaxation of IP-SC such that x is integral. In polynomial time, we can compute an integral solution $\hat{y}$ such that $\langle x, \hat{y} \rangle$ is feasible.*

2.1 The Residual LP

We now partition the vertex set and generate a modified instance of the problem.

- Let $M \subseteq V$ with $|M| = \omega$. For each $v \in M$, let m_v be a fixed integer multiplicity.
- Define $R = \{v \in V \setminus M : w_v \leq \min_{v \in M} w_v\}$.
- Define $Z = V \setminus (M \cup R)$.

We now give a modified linear program for each choice of M and guessed multiplicty for nodes in M. Denote its value by $\mathrm{OPT}_{\mathrm{res}}(M)$.

Our guessing step separates the expensive part of the solution from the residual rounding problem. If the optimal solution uses at most ω vertices, we can enumerate to find the exact solution. We may therefore assume that the optimal solution uses more than ω distinct vertices. In this case, we guess the set M of ω highest weight vertices and their multiplicities. Under the correct guess, every remaining optimal vertex is no more expensive than the least expensive guessed vertex, so we may safely exclude all more expensive unguessed vertices. Later, the guessed vertices also provide the budget for a set of rounded fractional vertices.

$$\begin{aligned}
\text{minimize} \quad & \sum_{v \in R} w_v x_v && && \text{(LP-RES}_{\mathrm{M}}\text{)} \\
\text{subject to} \quad & \sum_{v \in e} y_{e,v} \leq 1 && \forall e \in E \\
& \sum_{e \in E_g} \left(\sum_{v \in e} y_{e,v} \right) \geq \rho_g && \forall g \in \{1, \ldots, \omega\} \\
& \sum_{e \in \delta(v)} y_{e,v} \leq k_v x_v && \forall v \in V \\
& 0 \leq y_{e,v} \leq x_v && \forall v \in e \in E \\
& 0 \leq x_v \leq m && \forall v \in R \\
& x_v = m_v && \forall v \in M \\
& x_v = 0 && \forall v \in Z
\end{aligned}$$

For ease of notation, we will use $c_e := \sum_{v \in e} y_{e,v}$ to refer to the coverage of an edge.

▶ **Observation 7.** *Let* OPT^I *be the value of the optimal integral solution to IP-SC. There exists a set* $M \subseteq V$ *with selected multiplicites* $\{m_v\}_{v \in M}$ *such that the resulting* LP-RES$_{\mathrm{M}}$ *has integral solution of value* $\mathrm{OPT}^I_{\mathrm{res}}(M)$ *with*

$$\mathrm{OPT}^I_{\mathrm{res}}(M) + \sum_{v \in M} w_v m_v = \mathrm{OPT}^I .$$

The proof of the following lemma is somewhat technical and deferred to Subsection 2.3.

▶ **Lemma 8.** *For any feasible solution* $\langle x, y \rangle$ *to* LP-RES$_{\mathrm{M}}$*, we can construct a feasible solution* $\langle \hat{x}, \hat{y} \rangle$ *such that* $\hat{x}$ *is integral and has the following properties:*

1. *for vertices in* M*,*
$$\sum_{v \in M} w_v \hat{x}_v \leq f \sum_{v \in M} w_v x_v;$$
2. *for the vertices involved in* LP-RES$_{\mathrm{M}}$*,*
$$\sum_{v \in R} w_v \hat{x}_v \leq (f+1)\,\mathrm{OPT}_{\mathrm{res}}(M) + \sum_{v \in F} w_v,$$
where $F \subseteq R$ *with* $|F| \leq \omega$*.*

In order to complete the proof of Theorem 2, we need to show how the correct choice of $M \subseteq V$ eliminates the term $\sum_{v \in F} w_v$. Recall that for all $v \in F \subseteq R$ and all $v_j \in M$, $w_v \leq w_{v_j}$. By combining this with Observation 7, we get

$$\begin{aligned}
\sum_{v \in V} w_v \hat{x}_v &= \sum_{v \in M} w_v \hat{x}_v + \sum_{v \in R} w_v \hat{x}_v \\
&\leq f \sum_{v \in M} w_v m_v + (f+1)\,\mathrm{OPT}_{\mathrm{res}}(M) + \sum_{v \in F} w_v \\
&\leq f \sum_{v \in M} w_v m_v + (f+1)\left(\mathrm{OPT}^I - \sum_{v \in M} w_v m_v\right) + \sum_{v \in F} w_v \\
&= (f+1)\,\mathrm{OPT}^I + f \sum_{v \in M} w_v m_v - (f+1) \sum_{v \in M} w_v m_v + \sum_{v \in F} w_v \\
&= (f+1)\,\mathrm{OPT}^I - \sum_{v \in M} w_v m_v + \sum_{v \in F} w_v \\
&\leq (f+1)\,\mathrm{OPT}^I .
\end{aligned}$$

Thus our rounded solution, which is feasible with respect to both group and capacity constraints, exceeds the integral optimal solution by a factor of no more than $f+1$. There are $n^{O(\omega)}$ possible sets M, and each set has m^ω different sets of multiplicities. After checking all possibilities, we output the lowest-cost feasible solution; since this is the correct guess, its cost is bounded by Theorem 2.

Before we prove Lemma 8, we first describe a rounding scheme starting from the solution to LP-RES$_{\mathrm{M}}$.

2.2 The Rounding Scheme

We give a partition of V:

- $H := M \cup \left\{v \in R : x_v \geq \frac{1}{f+1}\right\}$
- $L := \left\{v \in R : 0 < x_v < \frac{1}{f+1}\right\}$

Any vertices with $x_v = 0$ are disregarded, and we set $\hat{x}_v = 0$. Now, we round the edge variables. If any edge is such that all vertices have $\hat{x}_v = 0$, that edge cannot be covered. For simplicity, we will remove such edges from consideration going forward (these will remain uncovered). For each $v \in L$, create an empty set L_v. Some edges may include vertices that are already rounded to 0. For clarity, let $L' = L \cup \{v \in V : \hat{x}_v = 0\}$.

The next step preserves the total coverage c_e of every edge. It only moves this coverage between endpoints such that the coverage of each edge is partitioned between vertices in H or given to one unique vertex in L.

1. $e \cap L' = \emptyset$: all coverage is handled by vertices in H. Set $\hat{y}_{e,v} = y_{e,v}$ for all $v \in e$.
2. $e \cap H = \emptyset$: all coverage is handled by vertices in L. Let $v_\ell = \arg\max_{v \in e \cap L} y_{e,v}$. Set $\hat{y}_{e,v_\ell} = c_e$ and $\hat{y}_{e,v} = 0 \;\forall\; v \in e \setminus v_\ell$. Add e to L_{v_ℓ}.
3. $e \cap H, e \cap L \neq \emptyset$: coverage is distributed between vertices in L and vertices in H.
   a. If $y_{e,v_\ell} \geq \frac{c_e}{f+1}$, let $v_\ell = \arg\max_{v \in e \cap L} y_{e,v}$. Set $\hat{y}_{e,v_\ell} = c_e$ and $\hat{y}_{e,v} = 0$ for all other $v \in e$. Add e to L_{v_ℓ}.
   b. Otherwise, let $c_e^\ell = \sum_{v \in e \cap L} y_{e,v}$ and $c_e^h = \sum_{v \in e \cap H} y_{e,v}$. Then, $\hat{y}_{e,v} = y_{e,v} + \frac{y_{e,v}}{c_e^h} c_e^\ell$ for $v \in e \cap H$ and $\hat{y}_{e,v} = 0$ for $v \in e \cap L$.

Note that, in case (3), if an endpoint in L is above the threshold $\frac{c_e}{f+1}$, it can absorb the entire edge with an $(f+1)$-fold increase; otherwise, the coverage provided by all endpoints in L is proportionately transfered to the endpoints in H. Therefore, the threshold is carefully chosen to work for both cases. Also observe that the sets L_v are disjoint, meaning that each edge has no more than one $v \in L$ that is responsible for its coverage. For now, we only round $v \in H$: $\hat{x}_v = \left\lceil \frac{f+1}{2} x_v \right\rceil$.

▶ **Lemma 9** (†). *For all $v \in H$,*

$$\sum_{e \in \delta(v)} \hat{y}_{e,v} \leq \frac{f+1}{2} k_v x_v \leq k_v \hat{x}_v.$$

Moreover, if $v \in H \cap R$, $\hat{x}_v \leq (f+1)x_v$, and if $v \in M$, $\hat{x}_v \leq f \cdot x_v$.

We are now covering some subset of edges in each group. For $g \in [\omega]$,

$$h_g := \sum_{v \in H} \sum_{e \in \delta(v) \cap E_g} \hat{y}_{e,v}.$$

In Lemma 9, we showed that the rounded multiplicities $\tilde{x}_v$ for $v \in H$ can cover all rounded fractional edges that they are responsible for. Now, we have to open some vertices in L to cover the remaining $\rho_g - h_g$ edges of each group.

2.3 The Configuration LP

Our final task is to round the vertices in L. For these vertices, we will only consider whether or not we will open one copy of each vertex. For $v \in L$ and $g \in [\omega]$, let $N_{v,g} = E_g \cap L_v$ be the set of edges of group g that v is responsible for, and let $\eta_{v,g} = |N_{v,g}|$.

Since each L_v is disjoint, some vertex v covering some edge $e \in L_v$ has no bearing on whether another vertex v' can cover some edge $e' \in L_{v'}$, even if v and v' are adjacent. Therefore, we consider the following question: if we were to open one copy of v, how many different ways could v cover up to k_v edges in L_v? With this framing, we simplify the remaining problem by rendering edges of the same group in each L_v indistinguishable.

A feasible configuration for a vertex $v \in L$ is a vector $A = (A_1, \ldots, A_\omega) \in \mathbb{Z}^\omega_{\geq 0}$ with $A_g \leq \eta_{v,g}$ for all $g \in [\omega]$ and $\sum_{g=1}^{\omega} A_g \leq k_v$. The coordinate A_g records how many owned edges of group g are served by one copy. The bounds $A_g \leq \eta_{v,g}$ and $\sum_g A_g \leq k_v$ enforce edge availability and capacity, respectively. Let $\mathcal{A}_v$ be the set of all feasible configurations for $v \in L$. We now solve this configuration LP:

$$\text{minimize} \quad \sum_{v \in L} w_v \sum_{A \in \mathcal{A}_v \setminus \mathbf{0}} \lambda_{v,A} \qquad \text{(LP-CONFIG)}$$

$$\text{subject to} \quad \sum_{A \in \mathcal{A}_v} \lambda_{v,A} = 1 \qquad \forall v \in L \qquad (1)$$

$$\sum_{v \in L} \sum_{A \in \mathcal{A}_v} A_g \lambda_{v,A} \geq \rho_g - h_g \qquad \forall g \in \{1, \ldots, \omega\} \qquad (2)$$

$$\lambda_{v,A} \geq 0 \qquad \forall v \in L, A \in \mathcal{A}_v$$

We consider the zero configuration $\mathbf{0}$ to mean that the vertex is not chosen, so each $\lambda_{v,\mathbf{0}}$ does not contribute to the cost.

We first need to connect the solution of LP-CONFIG to the solution of LP-RES$_{\text{M}}$ restricted to the vertices in L. In order to do this, we will construct a feasible solution to LP-CONFIG using the solution $\langle x, y \rangle$ restricted to vertices in L and edges in $\bigcup_{v \in L} L_v$ and the rounding scheme described in Subsection 2.2.

▶ **Lemma 10.** *LP-CONFIG has a feasible solution of cost no more than* $(f+1)\sum_{v\in L} w_v x_v$.

Proof. We start by considering the edges in the sets L_v for each $v \in L$. If an edge was rounded using case (2), we have

$$\hat{y}_{e,v} = c_e \leq f \cdot y_{e,v} \leq (f+1)x_v.$$

If it was rounded using case (3a), v must be such that $y_{e,v} \geq \frac{c_e}{f+1}$, so

$$\hat{y}_{e,v} = c_e \leq (f+1)y_{e,v} \leq (f+1)x_v.$$

We also observe that, because L contains only vertices with $x_v < \frac{1}{f+1}$, we must have $(f+1)x_v < 1$. Finally, by summing over edges in L_v, we have

$$\sum_{e\in L_v} c_e \leq (f+1)\sum_{e\in L_v} y_{e,v} \leq (f+1)k_v x_v$$

from the feasibility of the LP solution. This implies that

$$\sum_{e\in L_v} \frac{c_e}{(f+1)x_v} \leq k_v.$$

Consider the following set of vectors:

$$\mathcal{P}(d,k) = \left\{z \in \mathbb{R}^d : 0 \leq \|z\|_\infty \leq 1, \|z\|_1 \leq k\right\}.$$

This defines a concave set with integral extremal points. Assume that we ordered the edges of L_v. Then, the vector

$$z_v = \left(\frac{c_{e_1}}{(f+1)x_v}, \frac{c_{e_2}}{(f+1)x_v}, \dots\right)$$

is contained within $\mathcal{P}(|L_v|, k_v)$. Let $\mathcal{S}_v = \{S \subseteq L_v : |S| \leq k_v\}$. We can now use the set of indicators $\mathbf{1}_S$ of $S \in \mathcal{S}_v$ as a basis for z_v. Write $z_v = \sum_{S\in\mathcal{S}_v} \zeta_{v,S}\mathbf{1}_S$ for $\sum_{S\in\mathcal{S}_v} \zeta_{v,S} = 1$.

Next, we want to move from the vectors $\mathbf{1}_S$ to feasible edge configurations. Define the projection $\pi_v : \mathbb{R}^{|L_v|} \to \mathbb{R}^\omega$ such that column g is $\mathbf{1}_{N_{v,g}}$. Each $\pi_v\mathbf{1}_S$ is a feasible edge configuration. For each $A \in \mathcal{A}_v \setminus \mathbf{0}$, let $S_A = \{S \in \mathcal{S}_v : \pi_v S = A\}$. Define $\lambda_{v,A} = (f+1)x_v \sum_{S\in S_A} \zeta_{v,S}$ and $\lambda_{v,\mathbf{0}} = 1 - \sum_{A\neq\mathbf{0}} \lambda_{v,A}$. Therefore, λ satisfies the constraints (1).

Now, we look at the constraints (2). For one edge, $\sum_{S\ni e} \zeta_{v,S} = \frac{c_e}{(f+1)x_v}$. We know that the edges are sufficiently covered in the rounded solution, so

$$\begin{aligned}
\sum_{v\in L}\sum_{A\in\mathcal{A}_v} A_g\lambda_{v,A} &= \sum_{v\in L}\sum_{A\in\mathcal{A}_v} A_g \sum_{S\in S_A} \zeta_{v,S}(f+1)x_v \\
&= \sum_{v\in L}(f+1)x_v \sum_{S\in\mathcal{S}} \mathbf{1}_{N_{v,g}} \cdot \mathbf{1}_S \zeta_{v,S} \\
&= \sum_{v\in L}\sum_{S\in\mathcal{S}_v}\sum_{e\in S\cap N_{v,g}} (f+1)x_v\zeta_{v,S} \\
&= \sum_{v\in L}(f+1)x_v \sum_{e\in N_{v,g}}\sum_{S\ni e} \zeta_{v,S} \\
&= \sum_{v\in L}(f+1)x_v \sum_{e\in N_{v,g}} \frac{c_e}{(f+1)x_v} \\
&= \sum_{v\in L}\sum_{e\in N_{v,g}} c_e \geq \rho_g - h_g
\end{aligned}$$

Now that the projection defines a feasible solution to LP-CONFIG, we need to show that the objective function value is bounded:

$$\begin{aligned}\sum_{v\in L} w_v \sum_{A\in\mathcal{A}_v\setminus\mathbf{0}} \lambda_{v,A} &= \sum_{v\in L} w_v \sum_{A\in\mathcal{A}_v\setminus\mathbf{0}} (f+1)x_v \sum_{S\in S_A} \zeta_{v,S} \\ &= \sum_{v\in L} w_v(f+1)x_v \sum_{S\in\mathcal{S}\setminus\mathbf{0}} \zeta_{v,S} \\ &\le (f+1)\sum_{v\in L} w_v x_v.\end{aligned}$$

◀

Let $F \subseteq L$ be the set of vertices with fractional $\lambda_{v,A}$. While we also want to open this set of vertices, bounding the cost is made difficult by the lack of an integral configuration.

▶ **Lemma 11.** *Any optimal basic feasible solution λ to LP-CONFIG has $|F| \le \omega$.*

Proof. We start by turning constraints (2) into equalities. Our new LP is

$$\begin{aligned}\text{minimize} \quad & \sum_{v\in L} w_v \sum_{A\in\mathcal{A}_v\setminus\{0\}} \lambda_{v,A} && \\ \text{subject to} \quad & \sum_{A\in\mathcal{A}_v} \lambda_{v,A} = 1 && \forall v \in L \\ & \sum_{v\in L}\sum_{A\in\mathcal{A}} A_g\lambda_{v,A} = \rho_g - h_g + s_g && \forall g \in \{1,\dots,\omega\} \\ & \lambda_{v,A} \ge 0 && \forall v \in L, A \in \mathcal{A}_v \\ & s_g \ge 0 && \forall g \in \{1,\dots,\omega\}\end{aligned}$$

A solution to this LP has up to $|L|+\omega$ non-tight variables, meaning that up to $|L|+\omega$ of the variables $\lambda_{v,A}$ and s_g are positive. Constraints (1) ensures that at least one $\lambda_{v,A}$ per $v \in L$ is positive, and at least $|L|-\omega$ of those constraints has only one positive $\lambda_{v,A}$, so up to ω of these constraints can have fractional $\lambda_{v,A}$. ◀

We now have the machinery to prove Lemma 8. Recall the statement:

▶ **Lemma 8.** *For any feasible solution $\langle x, y\rangle$ to* LP-RES$_{\text{M}}$*, we can construct a feasible solution $\langle \hat{x}, \hat{y}\rangle$ such that $\hat{x}$ is integral and has the following properties:*

1. *for vertices in M,*

$$\sum_{v\in M} w_v\hat{x}_v \le f\sum_{v\in M} w_v x_v;$$

2. *for the vertices involved in* LP-RES$_{\text{M}}$*,*

$$\sum_{v\in R} w_v\hat{x}_v \le (f+1)\,\text{OPT}_{\text{res}}(M) + \sum_{v\in F} w_v,$$

where $F \subseteq R$ with $|F| \le \omega$.

Proof. For each $v \in L$, if $\lambda_{v,\mathbf{0}} = 1$, set $\hat{x}_v = 0$ and $\hat{y}_{e,v} = 0$ for all $e \in L_v$. Otherwise, set $\hat{x}_v = 1$. If $v \notin F$, there is one integral edge configuration. Select a set $N(A)$ consisting of A_g edges from each $N_{v,g}$, and set $\hat{y}_{e,v} = \mathbf{1}_{e\in N(A)}$. If $v \in F$, let $\mathcal{F}_v = \{A \in \mathcal{A}_v : \lambda_{v,A}\} > 0$. For each configuration $A \in \mathcal{F}_v$, define a set of edges $N(A)$. Then, set $\hat{y}_{e,v} = \sum_{A\in\mathcal{F}_v} \lambda_{v,A}\mathbf{1}_{e\in N(A)}$.

Since the solution λ is feasible for LP-CONFIG,

$$\sum_{v\in L}\sum_{e\in N_{v,g}}\hat{y}_{e,v} = \sum_{v\in L\setminus F}\sum_{e\in N_{v,g}}\mathbf{1}_{e\in N(A)} + \sum_{v\in F}\sum_{A\in\mathcal{F}_v}\sum_{e\in N_{v,g}}\lambda_{v,A}\mathbf{1}_{e\in N(A)}$$
$$= \sum_{v\in L}A_g + \sum_{v\in F}\sum_{A\in\mathcal{F}_v}\sum_{e\in N_{v,g}}\lambda_{v,A}A_g \geq \rho_g - h_g,$$

since the L_v are disjoint and no edge can be double-counted. Moreover, if $v \in L \setminus F$,

$$\sum_{e\in\delta(v)}\hat{y}_{e,v} = \sum_{g=1}^{\omega}A_g \leq k_v = k_v\hat{x}_v.$$

If $v \in F$,

$$\sum_{e\in\delta(v)}\hat{y}_{e,v} = \sum_{e\in\delta(v)}\sum_{A\in\mathcal{F}_v}\lambda_{v,A}\mathbf{1}_{e\in N(A)} = \sum_{e\in\delta(v)}\sum_{A\in\mathcal{F}_v}\lambda_{v,A}A_g \leq k_v = k_v\hat{x}_v.$$

From Lemma 9, the vertices in H respect the capacity constraints. Thus, we can conclude that $\langle\tilde{x},\tilde{y}\rangle$ over all $v \in L \cup H$ satisfies both the group and capacity constraints. We now bound the cost of the rounded vertices. We use Lemma 9. For part (1), we can sum over the bound on $\hat{x}_v$ for all vertices in M. For part (2),

$$\sum_{v\in R\cap H}w_v\hat{x}_v \leq (f+1)\sum_{v\in R\cap H}w_vx_v.$$

From Lemma 10,

$$\sum_{v\in L}w_v\hat{x}_v \leq (f+1)\sum_{v\in L}w_vx_v + \sum_{v\in F}w_v.$$

By combining these statements, we have

$$\sum_{v\in R}w_v\hat{x}_v \leq (f+1)\sum_{v\in R}w_vx_v + \sum_{v\in F}w_v = (f+1)\,\mathrm{OPT}_{\mathrm{res}}(M) + \sum_{v\in F}w_v,$$

which concludes the proof of Lemma 8. ◀

2.4 The Full Algorithm

We end this section with an overview of the algorithm and a discussion of the runtime.

1. Enumerate through every set of vertices of size at most ω and every integral multiplicity from 1 to m. If any set is feasible, take the cheapest feasible set. Otherwise, continue to the main algorithm. There are $(|V|+|E|)^{O(\omega)}$ such sets.
2. For every set $M \subseteq V$ with $|M| \leq \omega$ and every integral multiplicity from 1 to m, solve LP-RES$_{\mathrm{M}}$. There are $(|V|+|E|)^{O(\omega)}$ such sets.
   a. Round the solution using Lemma 9.
   b. Solve LP-CONFIG for the vertices in L. The LP has up to $|V|(|E|+1)^{\omega}$ variables and $|V|+\omega$ constraints.
   c. Round the fractional vertices using Lemmas 10 and 11.
   d. Continue with the lowest-cost feasible solution, which corresponds to the correct guess of M.
3. Find an integral edge assignment using Lemma 6. The network has $O(|V|+|E|+\omega)$ nodes and $O(|V|+|E|+\omega)$ arcs.

Each step requires time that is polynomial in the $|V|+|E|$, and we repeat each step a number of times that is polynomial in $|V|+|E|$. Since ω is assumed to be constant, the overall algorithm is polynomial in $|V|+|E|$. Note that this runtime is independent of the hyperedge size f. This completes the proof of Theorem 2.

3 Hard Capacitated Unweighted Partition Vertex Cover

In this section, we study the Hard Capacitated Unweighted Partition Vertex Cover problem in hypergraphs using iterative rounding. Before detailing the iterative steps, we define the LP relaxation using fractional vertex variables x and assignment variables y. These variables are constrained to satisfy the partition thresholds and vertex capacities while ensuring that no edge assignment exceeds the fractional status of its endpoints. This formulation minimizes the total fractional vertex count, establishing a baseline solution that informs the subsequent rounding iterations. The integer program closely resembles IP-SC, so we will not restate it. The formal relaxation is presented below:

$$\text{minimize} \quad \sum_{v \in V} x_v$$

subject to:

$$\sum_{v \in e} y_{e,v} \leq 1 \qquad \forall e \in E$$

$$\sum_{e \in E_g} \left(\sum_{v \in e} y_{e,v} \right) \geq \rho_g \qquad \forall g \in \{1, \ldots, \omega\}$$

$$y_{e,v} \leq x_v \qquad \forall v \in e \in E$$

$$\sum_{e \in \delta(v)} y_{e,v} \leq k_v x_v \qquad \forall v \in V$$

$$0 \leq x_v \leq 1 \qquad \forall v \in V$$

$$y_{e,v} \geq 0 \qquad \forall v \in e \in E$$

It is sufficient to find an integral solution for x; an integral assignment y can then be found efficiently via a max-flow computation. Our algorithm therefore focuses on finding an integral x with a bounded cost.

3.1 Algorithm

Our algorithm is based on an iterative rounding framework. The core idea is to repeatedly solve an LP relaxation, fix a subset of variables to integral values based on the solution, and then re-solve a smaller, updated LP on the remaining problem. This process is guided by a dynamic LP formulation that adapts to the decisions made in each step (see LP-IP).

We maintain several sets for the iterative rounding algorithm. Let D be the set of vertices whose final integral counts have been determined, and let T_v^g denote the set of edges of group g that have been definitively assigned to vertex v. We define $T = \bigcup_{v \in V} \bigcup_{g=1}^{\omega} T_v^g$ as the set of all covered edges. Based on the fractional solution from the previous step, we partition the remaining undecided vertices into two groups: $U = \{v \in V \setminus D \mid x_v^* \geq 1/f\}$, which contains vertices with high fractional values, and $W = \{v \in V \setminus D \mid 0 < x_v^* < 1/f\}$, which contains those with low fractional values. U is further divided into $U_> = \{v \in V \setminus D \mid x_v^* > 1/f\}$ and $U_= = \{v \in V \setminus D \mid x_v^* = 1/f\}$. Let $Z = \{v \in V \setminus D \mid x_v^* = 0\}$. Initially the sets T and D are empty.

$$\text{minimize} \quad \sum_{v \in V \setminus D} x_v \tag{LP-IP}$$

subject to:

$$\sum_{v \in e \setminus D} y_{e,v} \leq 1 - \sum_{v' \in e \cap D} \bar{y}_{e,v'} \qquad \forall e \in E \setminus T$$

$$\sum_{e \in E_g \setminus T} \sum_{v \in e \setminus D} y_{e,v} \geq \rho_g - \sum_{v \in V} |T_v^g| - \sum_{e \in E_g \setminus T} \left(\sum_{v' \in e \cap D} \bar{y}_{e,v'} \right) \qquad g \in \{1, \ldots, \omega\}$$

$$y_{e,v} \leq x_v \qquad \forall e \in E \setminus T, v \in e \setminus D$$

$$\sum_{e \in \delta(v) \setminus T} y_{e,v} \leq \left(k_v - \sum_{g \in \omega} |T_v^g| \right) x_v \qquad \forall v \in U_>$$

$$\sum_{e \in \delta(v) \setminus T} y_{e,v} \leq k_v x_v \qquad \forall v \in W$$

$$1/f \leq x_v \leq 1 \qquad \forall v \in U_>$$

$$0 \leq x_v \leq 1/f \qquad \forall v \in W$$

$$y_{e,v} \geq 0 \qquad \forall e \in E \setminus T, v \in e \setminus D$$

Note that there is no capacity constraint included for vertices in $U_=$ within LP-IP. This is because any vertex u entering $U_=$ is immediately processed by the algorithm (its value is fixed to $\bar{x}_u = 1$ and it is moved to the decided set D). Thus, it is no longer an active variable in the updated LP over $V \setminus D$, and omitting it does not cause the feasible region to grow. The complete iterative rounding framework is detailed in Algorithm 1. It formalizes the process of identifying which variables to fix, updating the problem state, and iterating until a fully integral solution is found.

Note that when a tight edge is found and permanently assigned (Lines 8-11), we intentionally do not fix x_u to 1. The variable x_u remains active and fractional in the updated LP so its residual capacity can be utilized for other incident edges, and to prevent a premature increase in the LP objective. Since $u \in U$, Lemma 6 guarantees its fractional value remains at least $1/f$, ensuring it will be safely rounded to 1 at the end of the algorithm.

3.2 Analysis

▶ **Lemma 12.** *Suppose a vertex u satisfies $x_u^* \geq 1/f$ at some iteration. Then $x_u^* \geq 1/f$ in all subsequent iterations as long as $u \notin D$. Consequently, the final integral value $\bar{x}_u$ will be at least 1.*

Proof. When $x_u^* \geq 1/f$, the vertex u is placed in the set U. We consider two cases based on its value.

- If $x_u^* = 1/f$, then $u \in U_=$. In the same iteration, the algorithm adds u to the set of decided vertices D and fixes its final integral value as $\bar{x}_u = 1$.
- If $x_u^* > 1/f$, then $u \in U_>$. For all subsequent iterations where u remains an undecided variable, the updated LP-IR includes the explicit constraint $x_u \geq 1/f$. This ensures its fractional value cannot drop below $1/f$.

Algorithm 1 Iterative Rounding Algorithm

Solve the initial LP Relaxation to get an extreme point solution $\langle x^*, y^* \rangle$.
$T_v^g \leftarrow \emptyset$ for all $v \in V$ (which implicitly sets $T = \emptyset$), $g \in \{1, \ldots, \omega\}$ and $D \leftarrow \emptyset$.
Initialize $U, U_>, U_=, W, Z$ based on $\langle x^*, y^* \rangle$.
repeat
  for $v \in Z$ **do** ▷ Handle zero-valued vertices
    Set $\bar{x}_v \leftarrow 0$.
    $D \leftarrow D \cup \{v\}$.
  end for
  if $y^*_{e,u} = x^*_u$ for some $u \in U$ and $e \in \delta(u) \setminus T$ **then** ▷ Assign tight edges
    Set $\bar{y}_{e,u} \leftarrow 1$, and $\bar{y}_{e,v} \leftarrow 0$ for $v \in e \setminus \{u\}$.
    $T_u^g \leftarrow T_u^g \cup \{e\}$ **for** $e \in E_g$.
  end if
  for $u \in U_=$ **do** ▷ Handle boundary-valued vertices
    Set $\bar{x}_u \leftarrow 1$.
    Set $\bar{y}_{e,u} \leftarrow y^*_{e,u}$ **for** $e \in \delta(u) \setminus T$. ▷ Fix edge assignments
    $D \leftarrow D \cup \{u\}$.
  end for
  Solve the updated LP-IR for a new solution $\langle x^*, y^* \rangle$.
  Update $U, U_>, U_=, W, Z$ based on the new $\langle x^*, y^* \rangle$.
until $U_= = \emptyset$ and $Z = \emptyset$ and no tight edges are found
for $v \notin D$ **do** ▷ Final rounding step
  Set $\bar{x}_v \leftarrow \lceil x^*_v \rceil$.
end for
return integral solution $\bar{x}$.

Eventually, either u enters $U_=$ (and is rounded to $\bar{x}_u = 1$) or the algorithm terminates with $u \notin D$. In the latter case, it is rounded up in the final step, yielding $\bar{x}_u = \lceil x^*_u \rceil \geq \lceil 1/f \rceil = 1$. In all possible outcomes, if u ever enters U, its final value $\bar{x}_u$ is at least 1. ◀

▶ **Lemma 13.** *The solution $\bar{x}$ returned by the algorithm is a feasible integral solution to the problem.*

Proof. The final solution $\bar{x}$ is integral by construction. Variables are set to integer values (0 or 1) during the loop, and the final step applies the ceiling function.

Next, the initial LP ensures that the group coverage ρ_g is satisfied fractionally. As the algorithm fixes variables and assigns edges to T, it reduces the coverage requirement for subsequent LPs. The final LP solution $\langle x^*, y^* \rangle$ satisfies the remaining coverage fractionally. The final rounding step, $\bar{x}_v = \lceil x^*_v \rceil$, only increases the available capacity, which means at least as many edges can be covered as in the fractional solution. Thus, the integral solution meets or exceeds the fractional coverage and satisfies all group coverage requirements.

Finally, we consider capacity constraints for each group of vertices:

- **For vertices in $U_>$:** The capacity used by any "tight" edges is explicitly subtracted and budgeted for in later steps. When the remaining vertices are rounded up, the new integer capacity is always greater than or equal to the fractional capacity required by the LP solution ($k_u \lceil x^*_u \rceil \geq k_u x^*_u$), ensuring the constraint holds.
- **For vertices in $U_=$:** When a vertex is rounded from $1/f$ to 1, the LP guarantees that its fractional edge assignments required a capacity of at most $(k_u - \sum_{g \in [\omega]} |T_u^g|)/f$. This is well within the newly allocated integer capacity of $(k_u - \sum_{g \in [\omega]} |T_u^g|)$, so the assignment is valid.
- **For vertices in W:** At the end of the algorithm, these vertices are rounded up from a value less than $1/f$ to 1. The LP solution's required capacity was less than the final integer capacity ($k_w x^*_w < k_w \cdot 1$), so the final assignment is feasible. ◀

▶ **Lemma 14.** *Let $\langle x^*, y^* \rangle$ be the extreme point solution upon which the algorithm terminates. $|W| \leq |U_1| + \omega$ where $U_1 = \{u \in U : x^*_u = 1\}$.*

Proof. The proof is by a counting argument on the variables and constraints of the final LP-IR. As an extreme point solution, it is uniquely defined by a system of linear equations formed by setting a subset of the LP's inequality constraints to be tight. A fundamental property of such solutions is that the number of variables must equal the number of linearly independent tight constraints.

At termination, the set of undecided vertices is $V \setminus D = U_> \cup W$. The algorithm's termination conditions ensure that for any $u \in U_>$, we have $x^*_u > 1/f$ and $y^*_{e,u} < x^*_u$. For any $w \in W$, we have $0 < x^*_w < 1/f$.

The variables in the final LP are the $\{x_v\}$ for undecided vertices and the associated $\{y_{e,v}\}$ assignment variables. The total number of variables is

$$|U_>| + |W| + \sum_{e \in E \setminus T} |e \setminus D| = |U_>| + |W| + \sum_{e \in E \setminus T} |e \cap U_>| + \sum_{e \in E \setminus T} |e \cap W|.$$

Next, we establish an upper bound on the number of linearly independent constraints that can be tight. For the **partition constraints**, **capacity constraints for $U_>$:**, and **upper bounds $x_u \leq 1$:** at most ω, $|U_>|$, and $|U_1|$ constraints can be tight.

The **constraints involving W vertices** include the capacity constraint for each $w \in W$ ($\sum y_{e,w} \leq k_w x_w$), the coupling constraints ($y_{e,w} \leq x_w$), and the non-negativity constraints ($y_{e,w} \geq 0$). The maximum number of constraints from this group is limited for two reasons. First, for any specific edge e, the constraints $y_{e,w} \geq 0$ and $y_{e,w} \leq x_w$ cannot both be tight simultaneously. This is because $x^*_w > 0$. Thus, for each edge, at most one of these two boundary constraints can be part of the basis. Second, the entire collection of constraints for vertex w is linearly dependent. For a given vertex w, if the constraints corresponding to $y_{e,w}$ are chosen to be all tight, the tightness of the capacity constraint is a direct consequence and is therefore redundant; it cannot add a new linearly independent equation to the basis. Due to these two properties, the maximum number of independent tight constraints associated with any single vertex w is $|\delta(w) \setminus T|$. Summing over all vertices in W establishes the total contribution of at most

$$\sum_{w \in W} |\delta(w) \setminus T| = \sum_{e \notin T} |e \cap W|$$

constraints to the basis.

Finally, in order to handle the **constraints involving $U_>$ vertices**, we do a per-edge analysis. For each edge $e \in E \setminus T$, we analyze the relationship between its edge coverage constraint and its non-negativity constraints for $U_>$ endpoints. Let C_e be the edge coverage constraint

$$\sum_{v \in e \setminus D} y_{e,v} \leq 1 - \sum_{v' \in e \cap D} \bar{y}_{e,v'},$$

and let $\{N_{e,u}\}$ be the set of non-negativity constraints: $y_{e,u} \geq 0$ for all $u \in e \cap U_>$.

▷ Claim. For any edge e with at least one endpoint in $U_>$, it is impossible for C_e and all constraints in $\{N_{e,u}\}$ to be tight simultaneously.

Proof. Assume for contradiction that for an edge e where $|e \cap U_>| \geq 1$, constraint C_e and all constraints $N_{e,u}$ are tight. The tightness of all $N_{e,u}$ implies $y_{e,u} = 0$ for all $u \in e \cap U_>$.

Thus, the edge's total coverage comes only from W vertices:

$$\sum_{v \in e \setminus D} y_{e,v} = \sum_{w \in e \cap W} y_{e,w}.$$

Since $y_{e,w} < 1/f$ for any $w \in W$, we have a strict upper bound:

$$\sum_{v \in e \setminus D} y_{e,v} < |e \cap W|/f.$$

The tightness of C_e means

$$\sum_{v \in e \setminus D} y_{e,v} = 1 - \sum_{v' \in e \cap D} \bar{y}_{e,v'}.$$

This right-hand side has a lower bound of $|e \setminus D|/f$. Combining these gives

$$|e \setminus D|/f \leq \sum_{v \in e \setminus D} y_{e,v} < |e \cap W|/f.$$

This implies

$$|e \cap U_>| + |e \cap W| < |e \cap W|,$$

which simplifies to $|e \cap U_>| < 0$, a contradiction. ◁

This claim implies that for each edge e, the set of $1 + |e \cap U_>|$ constraints (consisting of C_e and $\{N_{e,u}\}$) are linearly dependent when all are tight. Therefore, at most $|e \cap U_>|$ of them can be chosen for a linearly independent basis. The total number of tight constraints from this group, summed over all edges, is at most $\sum_{e \in E \setminus T} |e \cap U_>|$.

By equating the number of variables to the upper bound on the number of tight, linearly independent constraints, we have:

$$|U_>| + |W| + \sum_{e \in E \setminus T} |e \cap U_>| + \sum_{e \in E \setminus T} |e \cap W| \leq \omega + |U_>| + |U_1| + \sum_{e \in E \setminus T} |e \cap W| + \sum_{e \in E \setminus T} |e \cap U_>|.$$

Canceling the common terms from both sides yields the desired result. ◀

We can now return to the proof of Theorem 3.

▶ **Theorem 3.** *For any fixed $\epsilon > 0$, there exists an $(f+\epsilon)$-factor approximation algorithm for the* Hard Capacitated Unweighted Partition Vertex Cover *problem that runs in time $n^{O(\omega/\epsilon)}$.*

Proof. The feasibility of the integral solution $\bar{x}$ produced by the algorithm is guaranteed by Lemma 13. We now bound its cost.

Let OPT_{LP} be the optimal value of the initial LP relaxation. First, observe that the optimal value of the LP relaxation does not decrease between iterations. When rounding decisions are made in the main loop, the problem is modified by either fixing variables or enforcing stricter inequalities (e.g., $x_v \geq 1/f$ instead of $x_v \geq 0$). Because we shrink the feasible region of a minimization problem, the objective value cannot improve. The cost added to our integral solution during the main loop can be bounded by charging it to the fractional value removed from the LP. Specifically, when $x^*_u \geq 1/f$ is rounded to 1, the cost ratio is at most $1/(1/f) = f$.

Let S_L be the set of vertices finalized during the iterative loop and $S_F = U_> \cup W$ be the set of vertices rounded at the end. An accounting argument based on the above observation shows that the cost from vertices rounded during the loop, $|S_L|$, plus f times the cost of the final LP, $f \cdot \mathrm{OPT}_F$, is at most $f \cdot \mathrm{OPT}_{LP}$.

The cost from the final rounding step is $\mathrm{Cost}(S_F) = \sum_{v \in U_> \cup W} \lceil x^*_v \rceil = |U_>| + |W|$. By Lemma 14, this is bounded by $|U_>| + |U_1| + \omega$. For any $u \in U_>$, we have $1 < f \cdot x^*_u$. This is sufficient to show that, for $f \geq 2$,

$$|U_>| + |U_1| \leq f \cdot \sum_{u \in U_>} x^*_u \leq f \cdot \mathrm{OPT}_F \,.$$

Therefore, the cost of the final step is bounded as

$$\mathrm{Cost}(S_F) = |U_>| + |W| \leq f \cdot \mathrm{OPT}_F + \omega$$

Combining the costs from both phases gives the total cost

$$\mathrm{Cost}(\bar{x}) = |S_L| + \mathrm{Cost}(S_F) \leq |S_L| + f \cdot \mathrm{OPT}_F + \omega$$

Since $|S_L| + f \cdot \mathrm{OPT}_F \leq f \cdot \mathrm{OPT}_{LP}$, we have a solution with a total cost bounded by $f \cdot \mathrm{OPT}_{LP} + \omega$.

We enumerate all possible solutions of size $\kappa = 1, 2, \ldots, \lceil \omega/\epsilon \rceil$. This step takes $n^{O(\omega/\epsilon)}$ time. If a feasible solution is found, the first one encountered must be an optimal solution, since we are checking solution sizes in increasing order, and we are done.

Otherwise, if this process is completed without finding a feasible solution, we know that the optimal integral solution, OPT, has a cost greater than ω/ϵ. In this case, we run our iterative rounding algorithm. By the analysis above, it returns a feasible solution with a cost bounded by

$$\mathrm{Cost}(\bar{x}) \leq f \cdot \mathrm{OPT}_{LP} + \omega$$

Since $\mathrm{OPT}_{LP} \leq \mathrm{OPT}$ and we have established that $\mathrm{OPT} > \omega/\epsilon$, we can bound the cost of our solution as

$$\mathrm{Cost}(\bar{x}) < f \cdot \mathrm{OPT} + \epsilon \cdot \mathrm{OPT} = (f + \epsilon) \cdot \mathrm{OPT} \,.$$

This provides an $(f + \epsilon)$-approximation for the problem. ◀

4 Weighted Partition Edge Cover

We consider the following generalization from [3]:

▶ **Definition 15** (Weighted Budgeted Matching). *We are given a graph $G = (V, E)$ with a weight function $p : E \to \mathbb{R}_{\geq 0}$, a partition of the vertex set $V = V_1 \cup \cdots \cup V_\omega$, and a parameter ρ_g for each group. We want to find a minimum-weight matching such that at least ρ_g vertices from group V_g are matched.*

First, we note that the reduction between Partition-EC and Budgeted Matching in Lemma 5 [3] holds in the weighted case.

▶ **Lemma 16** ([3]). *If Weighted Budgeted Matching can be solved in time $T(n, m)$, then Partition-EC can be solved in time $T(2n, m + n) + O(m + n)$. Here n and m refer to the number of nodes and edges respectively.*

In order to be matched or covered at minimum cost, a vertex will pay for its least expensive adjacent edge, so we can modify the reduction by giving the edge between each vertex and its auxillary vertex this minimum adjacent weight. Moreover, we eliminate the dependence on ω.

Now, we give a simple reduction to the standard problem of finding a maximum weight matching in a graph. This allows us to generalize the problem to weighted graphs, improve the running time, and avoid the reduction to tropical matching completely. Our algorithm also admits the use of standard libraries for weighted matching.

1. Let M be large with $M > \sum_e p_e$.
2. Let H be a copy of G with the following vertex sets added. For each group g add a set V_g' of $|V_g| - \rho_g$ nodes. Add edges between all nodes in V_g' and V_g for each g. The weight of an edge e in H is $2M - p_e$ when $e \in E$, and the weight of an edge (x, y) where $x \in V_g', y \in V_g$ is M. See Figure 1 for an example
3. Find a maximum weight matching $\mathcal{M}_H$ in H using the algorithm by Gabow [17, 18] The matched edges induced by the nodes of G should be a minimum weight matching satisfying the group requirements. If the weight of the maximum weight matching is at most $(n-1)M$ then no solution exists.

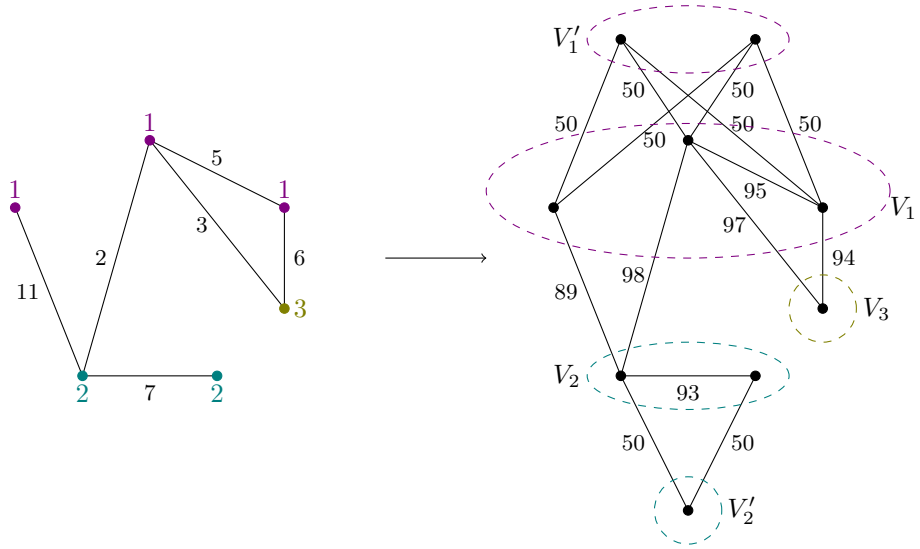


Figure 1 The reduction from Weighted Budgeted Matching to Maximum Matching for an instance with $\rho_1 = \rho_2 = \rho_3 = 1$. Here we let $M = 50$.

▶ **Lemma 17.** *If maximum-weight matching can be solved in time $T(n, m)$, then Weighted Budgeted Matching can be solved in time $T\left(2n, m + n^2\right)$.*

Proof. For each group g we add $\left|V_g'\right| \leq |V_g|$ vertices for a total of at most n additional vertices. We then add a complete bipartite graph with $\left|V_g'\right| \cdot |V_g|$ edges for each color, which totals at most n^2 additional edges. ◀

▶ **Lemma 18** (†). *In $\mathcal{M}_H$, at least ρ_g vertices from each group V_g are matched by edges from G.*

▶ **Lemma 19** (†). *The matching $\mathcal{M} = \mathcal{M}_H \cap E(G)$ with original edge weights has total weight equal to the optimum of the Weighted Budgeted Matching problem.*

To speed up the algorithm, we do a reduction to the problem of finding a max weight subgraph with upper bounds on degrees (f-factor). We contract the set V'_g into a single node V'_g with edges to all vertices in V_g of weight M and give a degree constraint of $|V_g| - \rho_g$ to V'_g. All nodes of V will have a degree constraint of 1. This new graph has only $n + \omega$ vertices, and at most n extra edges, preserving the sparsity of the graph. We now use the algorithm by Gabow [18] (see also [19]) which gives $O(nm + n^2 \log n)$ running time.

Lemmas 16, 17, 18 and 19 together imply Theorem 5.

5 Conclusion

While our algorithm achieves an $(f + \epsilon)$-approximation for the hard-capacitated problem, the algorithm for the weighted soft-capacitated (SC-PVC) variant has an approximation factor of $f+1$. Bridging this gap by designing an f-approximation algorithm for the weighted SC-PVC problem remains an intriguing open question. A natural question is about the dependence on ω in the run-time. Is it possible to derive an f approximation in time $O(f(\omega)n^c)$ so that the dependence on ω in the run-time is not $O(n^\omega)$? We already know that the problem is Set Cover hard for unbounded ω. The same runtime issue arises in the k-center problem as mentioned earlier (in related work).

Another interesting direction for future research is to develop an efficient combinatorial algorithm for the Partition Vertex Cover problem, even in the uncapacitated case. While such primal-dual methods exist for the related Partial Vertex Cover problem [34], developing one for the partition generalization is an open question.

For the distinct but related problem of maximizing edge coverage given a budget on the cardinality of the chosen vertex cover, Marx [33] shows how to obtain an FPT approximation scheme for unweighted simple graphs for $\omega = 1$. It will be interesting to obtain similar results for arbitrary ω.

References

1 Georg Anegg, Haris Angelidakis, Adam Kurpisz, and Rico Zenklusen. A technique for obtaining true approximations for k-center with covering constraints. *Math. Program.*, 192(1):3–27, 2022. doi:10.1007/S10107-021-01645-Y.

2 Tanvi Bajpai, Chandra Chekuri, and Pooja Kulkarni. Covering a few submodular constraints and applications. In Alina Ene and Eshan Chattopadhyay, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2025, August 11-13, 2025, Berkeley, CA, USA*, volume 353 of *LIPIcs*, pages 25:1–25:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.APPROX/RANDOM.2025.25.

3 Sayan Bandyapadhyay, Aritra Banik, and Sujoy Bhore. On colorful vertex and edge cover problems. *Algorithmica*, 85(12):3816–3827, 2023. doi:10.1007/S00453-023-01164-6.

4 Sayan Bandyapadhyay, Tanmay Inamdar, Shreyas Pai, and Kasturi R. Varadarajan. A constant approximation for colorful k-center. In Michael A. Bender, Ola Svensson, and Grzegorz Herman, editors, *27th Annual European Symposium on Algorithms, ESA 2019, September 9-11, 2019, Munich/Garching, Germany*, volume 144 of *LIPIcs*, pages 12:1–12:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ESA.2019.12.

5 Reuven Bar-Yehuda. Using homogenous weights for approximating the partial cover problem. In Robert Endre Tarjan and Tandy J. Warnow, editors, *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, 17-19 January 1999, Maryland, USA*, pages 71–75. ACM/SIAM, 1999. URL: http://dl.acm.org/citation.cfm?id=314500.314533.

6 Reuven Bar-Yehuda and Shimon Even. A linear-time approximation algorithm for the weighted vertex cover problem. *J. Algorithms*, 2(2):198–203, 1981. doi:10.1016/0196-6774(81)90020-1.

7 Reuven Bar-Yehuda and Shimon Even. A local-ratio theorem for approximating the weighted vertex cover problem. In Manfred Nagl and Jürgen Perl, editors, *Proceedings of the WG '83, International Workshop on Graphtheoretic Concepts in Computer Science, June 16-18, 1983, Haus Ohrbeck, Germany*, pages 17–28. Universitätsverlag Rudolf Trauner, Linz, 1983.

8 Reuven Bar-Yehuda, Guy Flysher, Julián Mestre, and Dror Rawitz. Approximation of partial capacitated vertex cover. *SIAM J. Discret. Math.*, 24(4):1441–1469, 2010. doi:10.1137/080728044.

9 Suman Kalyan Bera, Shalmoli Gupta, Amit Kumar, and Sambuddha Roy. Approximation algorithms for the partition vertex cover problem. *Theor. Comput. Sci.*, 555:2–8, 2014. doi:10.1016/J.TCS.2014.04.006.

10 Nader H. Bshouty and Lynn Burroughs. Massaging a linear programming solution to give a 2-approximation for a generalization of the vertex cover problem. In Michel Morvan, Christoph Meinel, and Daniel Krob, editors, *STACS 98, 15th Annual Symposium on Theoretical Aspects of Computer Science, Paris, France, February 25-27, 1998, Proceedings*, volume 1373 of *Lecture Notes in Computer Science*, pages 298–308. Springer, 1998. doi:10.1007/BFB0028569.

11 Wang Chi Cheung, Michel X. Goemans, and Sam Chiu-wai Wong. Improved algorithms for vertex cover with hard capacities on multigraphs and hypergraphs. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1714–1726. SIAM, 2014. doi:10.1137/1.9781611973402.124.

12 Julia Chuzhoy and Joseph Naor. Covering problems with hard capacities. *SIAM J. Comput.*, 36(2):498–515, 2006. doi:10.1137/S0097539703422479.

13 Rajni Dabas, Samir Khuller, and Emilie Rivkin. Weighted partition vertex and edge cover. *CoRR*, abs/2508.13055, 2025. doi:10.48550/arXiv.2508.13055.

14 Rajni Dabas, Samir Khuller, and Emilie Rivkin. Serving clients fairly: On facility location and k-median with fair outliers. In Huijia (Rachel) Lin, editor, *7th Symposium on Foundations of Responsible Computing, FORC 2026, Harvard University, Cambridge, MA, USA, June 3-5, 2026*, volume 368 of *LIPIcs*, pages 9:1–9:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. doi:10.4230/LIPIcs.FORC.2026.9.

15 Jennifer Dill and Nathan McNeil. Are shared vehicles shared by all? a review of equity and vehicle sharing. *Journal of Planning Literature*, 36 (1):5–30, 2020. doi:10.1177/0885412220966732.

16 Irit Dinur and David Steurer. Analytical approach to parallel repetition. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 624–633. ACM, 2014. doi:10.1145/2591796.2591884.

17 Harold N. Gabow. Data structures for weighted matching and nearest common ancestors with linking. In David S. Johnson, editor, *Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms, 22-24 January 1990, San Francisco, California, USA*, pages 434–443. SIAM, 1990. URL: http://dl.acm.org/citation.cfm?id=320176.320229.

18 Harold N. Gabow. Data structures for weighted matching and extensions to b-matching and f-factors. *ACM Trans. Algorithms*, 14(3):39:1–39:80, 2018. doi:10.1145/3183369.

19 Harold N. Gabow and Piotr Sankowski. Algorithms for weighted matching generalizations II: f-factors and the special case of shortest paths. *SIAM J. Comput.*, 50(2):555–601, 2021. doi:10.1137/16M1106225.

20 Rajiv Gandhi, Eran Halperin, Samir Khuller, Guy Kortsarz, and Aravind Srinivasan. An improved approximation algorithm for vertex cover with hard capacities. *J. Comput. Syst. Sci.*, 72(1):16–33, 2006. doi:10.1016/J.JCSS.2005.06.004.

21 Rajiv Gandhi, Samir Khuller, and Aravind Srinivasan. Approximation algorithms for partial covering problems. In Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen, editors, *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings*, volume 2076 of *Lecture Notes in Computer Science*, pages 225–236. Springer, 2001. doi:10.1007/3-540-48224-5_19.

22 Fănică Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM Journal on Computing*, 1(2):180–187, 1972. doi:10.1137/0201013.

23 Sudipto Guha, Refael Hassin, Samir Khuller, and Einat Or. Capacitated vertex covering. *J. Algorithms*, 48(1):257–270, 2003. doi:10.1016/S0196-6774(03)00053-1.

24 Dorit S. Hochbaum. Approximation algorithms for the set covering and vertex cover problems. *SIAM J. Comput.*, 11(3):555–556, 1982. doi:10.1137/0211045.

25 Eunpyeong Hong and Mong-Jen Kao. Approximation algorithm for vertex cover with multiple covering constraints. *Algorithmica*, 84(1):1–12, 2022. doi:10.1007/s00453-021-00885-w.

26 David Ingold and Spencer Soper. Amazon doesn't consider the race of its customers. should it?, April 2016. Accessed: 2026-01-20. URL: https://www.bloomberg.com/graphics/2016-amazon-same-day/.

27 Xinrui Jia, Kshiteej Sheth, and Ola Svensson. Fair colorful k-center clustering. *Math. Program.*, 192(1):339–360, 2022. doi:10.1007/S10107-021-01674-7.

28 Mong-Jen Kao. Iterative partial rounding for vertex cover with hard capacities. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, January 16-19*, pages 2638–2653. SIAM, 2017. doi:10.1137/1.9781611974782.174.

29 Mong-Jen Kao, Jia-Yau Shiau, Ching-Chi Lin, and D. T. Lee. Tight approximation for partial vertex cover with hard capacities. *Theor. Comput. Sci.*, 778:61–72, 2019. doi:10.1016/J.TCS.2019.01.027.

30 Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972. doi:10.1007/978-1-4684-2001-2_9.

31 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-ϵ. *J. Comput. Syst. Sci.*, 74(3):335–349, May 2008. doi:10.1016/j.jcss.2007.06.019.

32 Xiaofei Liu and Weidong Li. Approximation algorithm for prize-collecting hypergraph vertex cover with fairness constraints. In *Computing and Combinatorics (COCOON 2025)*, volume 15984 of *Lecture Notes in Computer Science*, pages 41–52, Singapore, 2026. Springer. doi:10.1007/978-981-95-0218-9_4.

33 Dániel Marx. Parameterized complexity and approximation algorithms. *Comput. J.*, 51(1):60–78, 2008. doi:10.1093/COMJNL/BXM048.

34 Julián Mestre. A primal-dual approximation algorithm for partial vertex cover: Making educated guesses. *Algorithmica*, 55(1):227–239, 2009. doi:10.1007/S00453-007-9003-Z.

35 George L. Nemhauser and Leslie E. Trotter Jr. Vertex packings: Structural properties and algorithms. *Math. Program.*, 8(1):232–248, 1975. doi:10.1007/BF01580444.

36 Barna Saha and Samir Khuller. Set cover revisited: Hypergraph cover with hard capacities. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, volume 7391 of *Lecture Notes in Computer Science*, pages 762–773. Springer, 2012. doi:10.1007/978-3-642-31594-7_64.

37 Sam Chiu-wai Wong. Tight algorithms for vertex cover with hard capacities on multigraphs and hypergraphs. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, 16-19*, pages 2626–2637. SIAM, 2017. doi:10.1137/1.9781611974782.173.

A Remaining Proofs from Section 2

▶ **Lemma 6** (†). *Let $\langle x, y\rangle$ be a feasible solution to the linear relaxation of IP-SC such that x is integral. In polynomial time, we can compute an integral solution $\hat{y}$ such that $\langle x, \hat{y}\rangle$ is feasible.*

Proof. We present the construction for completeness. Consider the input instance and a solution $\langle x, y\rangle$ with x integral. We construct a network flow instance F. The nodes of F are $s \cup E \cup (V \cup \{U_1, \ldots, U_\omega\}) \cup t$. The source and sink are s and t, respectively. The nodes $U_1, \ldots, U_\omega$ correspond to the uncovered sets for each group. We define the arcs and capacities:

- (s, e) for each $e \in E$ with capacity 1;
- (e, v) for each $v \in e$ with capacity 1;
- (e, U_g) where $e \in E_g$ with capacity 1;
- (v, t) for each $v \in V$ with capacity $k_v x_v$;
- (U_g, t) for each $g = 1, \ldots, \omega$ with capacity $|E_g| - \rho_g$.

We know that $\langle x, y\rangle$ is feasible, so there must be a flow of $|E|$ in F from s to t. The capacity on each arc is integral, so there exists and integral flow of value $|E|$ from s to t. Define $\hat{y}_{e,v} = 1$ (meaning that $\varphi(e) = v$) if the flow (e, v) is 1 and $\hat{y}_{e,v} = 0$ otherwise. If an edge goes unassigned, meaning that the flow from e to U_g is 1, $\varphi(e) = \perp$. Then, $\langle x, \hat{y}\rangle$ is a feasible integral solution. ◀

▶ **Lemma 9** (†). *For all $v \in H$,*

$$\sum_{e \in \delta(v)} \hat{y}_{e,v} \leq \frac{f+1}{2} k_v x_v \leq k_v \hat{x}_v.$$

Moreover, if $v \in H \cap R$, $\hat{x}_v \leq (f+1)x_v$, and if $v \in M$, $\hat{x}_v \leq f \cdot x_v$.

Proof. We only consider vertices affected by rounding cases (1) and (3b). For (1), the value of the edge variable is unchanged. In the case of (3b), we know that for all $v \in e \cap L$, $y_{e,v} < \frac{c_e}{f+1}$. We consider the worst possible case for rounding here. An endpoint in H handles $1 + \frac{c_e^\ell}{c_e^h}$ times more edge coverage, which increases as the mass c_e^ℓ increases. This added responsibility is therefore maximized when the edge contains $f - 1$ endpoints in L and a single endpoint in $v_h \in H$. Hence it suffices to analyze this worst case. Then,

$$\hat{y}_{e,v_h} = c_e = y_{e,v_h} + \sum_{v \in e \cap L} y_{e,v} < y_{e,v_h} + (f-1)\frac{c_e}{f+1}.$$

We can then conclude that

$$\hat{y}_{e,v_h} = c_e < \frac{f+1}{2} y_{e,v_h}.$$

Taking the sum over edges incident to v gives us the first inequality.

Now, we bound $\hat{x}_v$. For $v \in H \cap R$, we consider two cases. If $\frac{1}{f+1} \leq x_v \leq \frac{2}{f+1}$, then

$$\hat{x}_v = \left\lceil \frac{f+1}{2} x_v \right\rceil = 1 \leq (f+1)x_v.$$

Otherwise, we have $x_v > \frac{2}{f+1}$. In that case,

$$\hat{x}_v = \left\lceil \frac{f+1}{2} x_v \right\rceil < \frac{f+1}{2} x_v + 1 < (f+1)x_v.$$

For $v \in M$, $x_v \in \mathbb{N}$. Therefore,

$$\hat{x}_v = \left\lceil \frac{f+1}{2} x_v \right\rceil \leq \lceil f \cdot x_v \rceil = f \cdot x_v.$$

where the first inequality follows simply because $\frac{f+1}{2} \leq f$ for $f \geq 2$ and last equality follows because both f and x_v are integers. ◀

B Remaining Proofs from Section 4

▶ **Lemma 18** (†). *In $\mathcal{M}_H$, at least ρ_g vertices from each group V_g are matched by edges from G.*

Proof. Let $\mathcal{M}^* \subseteq G$ be an optimal solution to the budgeted matching problem. We now construct a matching $\mathcal{M}^*_H$ in H by including all edges from $\mathcal{M}^*$, and for each vertex in V_g not matched in $\mathcal{M}^*$, adding an auxiliary edge to a unique free node in V'_g.

Let k^*_g be the number of vertices in V_g matched by $\mathcal{M}^*$. Since $k^*_g \geq \rho_g$, we have, $|V_g| - k^*_g \leq |V_g| - \rho_g = |V'_g|$, so there are enough auxiliary vertices in V'_g to complete the matching without conflict. Thus, $\mathcal{M}^*_H$ is a feasible matching in H that matches all the vertices in V_g for every g. This also ensures that the cost of the maximum matching in H is strictly greater than $(n-1)M$ since each matched node in V contributes M to the weight of the matching, and even after subtracting the weight of a few edges we have a solution of weight between nM and $(n-1)M$. Any matching that does not match all nodes of G (that also belong to H) cannot have weight larger than $(n-1)M$.

Now consider the maximum weight matching $\mathcal{M}_H$ in H. Suppose, for contradiction, that $\mathcal{M}_H$ matches fewer than ρ_g vertices from some group V_g via edges from G. The number of vertices matched by auxiliary edges is at most $|V'_g| = |V_g| - \rho_g$. Therefore, at least one vertex $u \in V_g$ must be unmatched by $\mathcal{M}_H$. Therefore, for large M, $\mathcal{M}^*_H$ is better than $\mathcal{M}_H$, which is a contradiction. ◀

▶ **Lemma 19** (†). *The matching $\mathcal{M} = \mathcal{M}_H \cap E(G)$ with original edge weights has total weight equal to the optimum of the Weighted Budgeted Matching problem.*

Proof. We assume $n = |V|$ is even. This is perfectly fine because the reduction from Partition Edge Cover in Lemma 15 yields a Budgeted Matching instance with exactly $2n$ vertices, ensuring it is always an even-size instance. Let $\mathcal{M}^* \subseteq G$ be an optimal solution to the budgeted matching problem with total weight OPT. As in proof of Lemma 18, we can extend $\mathcal{M}^*$ to a matching $\mathcal{M}^*_H$ in H by adding auxiliary edges of weight M to unmatched vertices in V. Since all $n = |V|$ vertices in V are matched in $\mathcal{M}^*_H$, and exactly $n - 2|\mathcal{M}^*|$ auxiliary edges are used, the total weight of $\mathcal{M}^*_H$ is:

$$\sum_{e \in \mathcal{M}^*_H} p^H_e = \sum_{e \in \mathcal{M}^*} (2M - p_e) + (n - 2|\mathcal{M}^*|)M = nM - \text{OPT}.$$

Let $\mathcal{M}_H$ be a maximum weight matching in H, and let $\mathcal{M} = \mathcal{M}_H \cap E(G)$. Note that $\mathcal{M}_H$ also matches all n vertices in V. Therefore,

$$\sum_{e \in \mathcal{M}_H} p^H_e = \sum_{e \in \mathcal{M}} (2M - p_e) + (n - 2|\mathcal{M}|)M = nM - \sum_{e \in \mathcal{M}} p_e.$$

Since $\mathcal{M}_H$ maximizes the weight,

$$\sum_{e \in \mathcal{M}_H} p^H_e \geq \sum_{e \in \mathcal{M}^*_H} p^H_e \Rightarrow \sum_{e \in \mathcal{M}} p_e \leq \text{OPT}.$$

By Lemma 18, $\mathcal{M}$ is a feasible solution to the budgeted matching problem, so equality must hold and $\mathcal{M}$ is optimal. ◀

A Bad Example for Jain's Iterative Rounding Theorem for the Cover Small Cuts Problem

Miles Simmons
Department of Combinatorics and Optimization, University of Waterloo, Canada

Ishan Bansal
Amazon, Bellevue, WA, USA

Joe Cheriyan
Department of Combinatorics and Optimization, University of Waterloo, Canada

Abstract

The Cover Small Cuts problem is a key problem in Network Design. In the Cover Small Cuts problem, we are given a capacitated (undirected) graph $G = (V, E, u)$ and a threshold value λ, as well as a set of links L with end-nodes in V and a non-negative cost for each link $\ell \in L$; the goal is to find a minimum-cost set of links such that each non-trivial cut of capacity less than λ is covered by a link.

Jain's iterative rounding theorem is a well-known result in the area of approximation algorithms and, more broadly, in combinatorial optimization. The theorem asserts that LP relaxations of several problems in network design and combinatorial optimization have the following key property: for every basic feasible solution x there exists a variable x_e that has value at least a constant (e.g., $x_e \geq \frac{1}{2}$).

We construct an example showing that this property fails to hold for the standard LP relaxation of the Cover Small Cuts problem. This indicates that the polyhedron of feasible solutions to the LP (for Cover Small Cuts) differs in an essential way from the polyhedrons associated with several problems in combinatorial optimization. Moreover, our example shows that a direct application of Jain's iterative rounding algorithm does not give an $O(1)$ approximation algorithm for Cover Small Cuts.

We mention that Bansal et al. [6] showed that the WGMV primal-dual algorithm, due to Williamson et al. [19], applied to the same standard LP relaxation achieves approximation ratio 16 for the Cover Small Cuts problem. That is, the WGMV primal-dual algorithm, applied to the same LP relaxation, finds an integer solution of cost ≤ 16 times the optimal value of the LP.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases approximation algorithms, capacitated network design, covering small cuts, edge-connectivity of graphs, f-connectivity problem, iterative rounding method, primal-dual method

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.9

Category APPROX

Related Version *Previous Version*: `https://doi.org/10.48550/arXiv.2504.13105`

Funding *Ishan Bansal*: This work is external and does not relate to the position at Amazon.
Joe Cheriyan: Supported in part by NSERC grant RGPIN-2024-04473.

Acknowledgements We thank the anonymous reviewers and the Approx 2026 PC for their comments. We are grateful to Sharat Ibrahimpur for reading a preliminary version, and for valuable comments.

1 Introduction

The Cover Small Cuts problem has emerged as a key problem in Network Design since the publication of the first major result in ICALP 2023, see [5, 6]. In the Cover Small Cuts problem, we are given a capacitated (undirected) graph $G = (V, E, u)$ and a threshold value λ, as well as a set of links L with end-nodes in V and a non-negative cost c_ℓ for each link $\ell \in L$; the goal is to find a minimum-cost set of links such that each non-trivial cut of


© Miles Simmons, Ishan Bansal, and Joe Cheriyan;
licensed under Creative Commons License CC-BY 4.0

capacity less than λ is covered by a link. In more detail, a subset $F \subseteq L$ of links is said to *cover* a node-set S if there exists a link $\ell \in F$ with exactly one end-node in S, and the objective is to find a minimum-cost $F \subseteq L$ that covers each nonempty $S \subsetneq V$ such that $u(\delta_E(S)) < \lambda$. Bansal et al. [6] formulated this problem, and they showed that the WGMV primal-dual algorithm, due to Williamson et al. [19], achieves approximation ratio 16 for this problem. Later, Bansal [2, 3] and Nutov [15] proved that the same algorithm achieves approximation ratio 6. Simmons et al. [18, 17] improved the approximation ratio to 5. Recently, Nutov [16] constructed an example of the Cover Small Cuts problem such that the WGMV primal-dual algorithm finds a solution of cost $(5-\epsilon)\text{OPT}$, where OPT denotes the cost of an optimal (integral) solution and $\epsilon > 0$ is an arbitrarily small number. This shows that the 5-approximation analysis of [18, 17] is tight We mention that the same problem is called by other names in the recent literature, e.g., AugSmallCuts [6], Small Cut Augmentation [3], Small Cuts Cover [15], etc.

We may use abbreviations for some standard terms, e.g., we may use Cover Small Cuts as an abbreviation for "the Cover Small Cuts problem".

Given a graph $G = (V, E)$ whose edges have both costs and capacities, a fundamental task in network design is to find a spanning subgraph of minimum cost that satisfies some specified edge-connectivity requirements. *Capacitated network design* refers to the more general (and more challenging) setting where edges have non-uniform capacities. This topic has attracted intense research efforts over the last few years; see below for discussion and citations. In the context of algorithmic research and combinatorial optimization, the Cap-k-ECSS problem sits in the core of capacitated network design. In this problem, we are given an (undirected) graph $G = (V, E)$, where each edge has a non-negative cost and a positive integer capacity, and the algorithmic goal is to find a minimum-cost spanning subgraph such that every non-trivial cut has capacity k or more. Let $n = |V|$ and let u_{min} (respectively, u_{max}) denote the minimum (respectively, maximum) capacity of an edge; assume that $u_{max} \leq k$. One of the earliest approximation algorithms for Cap-k-ECSS is due to Goemans et al. [10], and it achieves an approximation ratio of $\min\{2k, |E|\}$. Over the last three decades, there have been several advances, and the best approximation ratio known is $\min(O(\log n), O(\log(k/u_{min})), k, 2u_{max})$, due to, respectively, Chakrabarty et al. [8], Bansal et al. [4], and Boyd et al. [7]. Let us mention that there are no known hardness results that rule out the possibility of better asymptotic approximation ratios. Most of the advances of this decade on Cap-k-ECSS are based on $O(1)$ approximation algorithms for Cover Small Cuts.

For the benefit of readers who are not familiar with the papers cited above, let us sketch a simple reduction from Cap-k-ECSS to Cover Small Cuts that incurs a multiplicative factor of $\lceil \frac{k}{u_{min}} \rceil$ in the approximation ratio, see [6, section 5] for further details, and see section 1.3 below for our notation. The algorithm is as follows: Initialize the "solution" edge-set F to be the empty set; during the execution of the algorithm, $G_F = (V, F)$ is our current "solution graph." While the minimum capacity of a cut $\delta_F(S)$, $\emptyset \neq S \subsetneq V$, in (V, F) is less than k, run an α-approximation algorithm for Cover Small Cuts with input $G_F = (V, F)$ and with link-set $L := E - F$, to augment all cuts $\delta_F(S)$, $\emptyset \neq S \subsetneq V$, with $u(\delta_F(S)) < k$ and obtain a valid augmentation $F' \subseteq L$. Update F by adding F', that is, $F := F \cup F'$. On exiting the while loop, output the edge-set F. Bansal et al. [6, Theorem 1.3] show that the cost of F is $\leq \alpha \cdot \lceil \frac{k}{u_{min}} \rceil \cdot \text{OPT}$, where OPT denotes the optimal cost for the Cap-k-ECSS instance.

1.1 Jain's iterative rounding method and the f-connectivity problem

Jain's iterative rounding method [12] and its extensions such as iterative relaxation [14] form a key theoretical tool for proving approximation guarantees, as well as characterizing optimal solutions of well-known (poly-time solvable) problems in combinatorial optimization.

Several connectivity augmentation problems can be formulated in a general framework called f-connectivity. In this problem, we are given an undirected graph $G = (V, E)$ on n nodes with nonnegative costs $c \in \mathbb{Q}^E_{\geq 0}$ on the edges and a requirement function $f : 2^V \to \mathbb{Z}_{\geq 0}$ on subsets of nodes. The algorithmic goal is to find an edge-set $J \subseteq E$ with minimum cost $c(J) := \sum_{e \in J} c_e$ such that for all cuts $\delta(S)$, $S \subseteq V$, we have $|\delta(S) \cap J| \geq f(S)$. A function f is called *weakly supermodular* if $f(V) = 0$, and for all $A, B \subseteq V$, either $f(A) + f(B) \leq f(A - B) + f(B - A)$, or $f(A) + f(B) \leq f(A \cap B) + f(A \cup B)$. The following is an LP relaxation for the f-connectivity problem.

$$\begin{aligned} \text{min} \quad & \sum_{e \in E} c_e x_e && && \text{(LP: f-connectivity)} \\ \text{subject to:} \quad & \sum_{e \in E \cap \delta(S)} x_e \geq f(S) && \forall\, S \subset V \\ & 0 \leq x_e \leq 1 && \forall\, e \in E. \end{aligned}$$

Assuming that the function f is weakly supermodular, integral, and has a positive value for some $S \subset V$, Jain [12] presented a 2-approximation algorithm for the f-connectivity problem, based on the following key result.

▶ **Theorem 1** (Jain [12, Theorem 3.1]). *Assume that the function f is weakly supermodular, integral, non-negative and non-zero. Then, in any basic feasible solution x to the LP relaxation, for at least one edge, e, x_e is at least $1/2$.*

The Jain property

For the sake of notational convenience, let us define a particular LP relaxation of a combinatorial problem to have the *Jain property* if there exists a constant c_0 such that every basic feasible solution x of the LP relaxation has an entry x_e such that $x_e \geq 1/c_0$ (i.e., the infinity norm of x is lower-bounded by a constant).

1.2 Our main result: A bad extreme point for the LP relaxation of Cover Small Cuts

We construct an example showing that the Jain property fails to hold for the standard LP relaxation of the Cover Small Cuts problem, and we mention that the WGMV primal-dual algorithm applied to the *same* LP relaxation finds an integer solution of cost $\leq 6 \cdot \text{LP}_{\text{opt}}$. This indicates that the polyhedron of feasible solutions to the LP relaxation (for Cover Small Cuts) differs in an essential way from the polyhedrons associated with several problems in combinatorial optimization. Moreover, our example shows that a direct application of Jain's iterative rounding algorithm does not give an $O(1)$ approximation algorithm for Cover Small Cuts.

Consider an instance $G = (V, E, u)$, λ of the Cover Small Cuts problem. Let $\mathcal{C} = \{\emptyset \neq S \subsetneq V : u(\delta_E(S)) < \lambda\}$. Then we have the following covering LP relaxation of the problem.

$$\begin{aligned} \text{LP}_{\text{opt}} := \min \quad & \sum_{f \in L} c_f x_f && \text{(LP: Cover Small Cuts)} \\ \text{subject to:} \quad & \sum_{f \in L \cap \delta(S)} x_f \geq 1 && \forall\ S \in \mathcal{C} \\ & 0 \leq x_f && \forall\ f \in L. \end{aligned}$$

▶ Remark 2. Let $f_{\text{CSC}} : 2^V \to \{0, 1\}$ be the requirement function for Cover Small Cuts; thus, for $S \subseteq V$, $f_{\text{CSC}}(S) = 1$ iff $S \in \mathcal{C}$. Although f_{CSC} is *not* weakly supermodular, f_{CSC} is a pliable function that satisfies property (γ), see [6] for definitions, and the $O(1)$ approximation analysis of [6] relies on this.

▶ **Theorem 3.** *Let $k \geq 4$ be an even integer. There is an instance of the* Cover Small Cuts *problem $G = (V, E), L, u$ such that the LP relaxation* (LP: Cover Small Cuts) *has a basic feasible solution x^* such that every positive variable of x^* has value $1/k$.*

Discussion and contrast with previous results

Jain's iterative rounding theorem [12] (see Theorem 1 above) asserts that for the LP relaxation of f-connectivity, whenever f is weakly supermodular, every basic feasible solution x has at least one variable $x_e \geq \frac{1}{2}$. This Jain property is the lever behind Jain's 2-approximation algorithm and has been applied widely in network design; see [14] for a comprehensive treatment. A natural question is whether the Jain property extends beyond the weakly supermodular setting, in particular, to LP relaxations of problems with *node-connectivity* requirements. The short answer is no, and the previous literature shows that there are (at least) two distinct reasons.

The first, and fundamental, reason comes from the area of hardness of approximation. Kortsarz, Krauthgamer, and Lee [13] proved that the node-connectivity Survivable Network Design Problem (NC-SNDP) admits no polynomial-time $O(1)$-approximation algorithm unless NP $\subseteq$ DTIME($n^{O(\log \log n)}$). Since the natural LP relaxation of NC-SNDP can be solved in polynomial time via a separation oracle based on max-flow computations, this hardness result implies – by a direct contrapositive argument – that the Jain property cannot hold for that LP: if every basic feasible solution x had some variable $x_e \geq 1/c_0$ for a fixed positive constant c_0, then iterating Jain's rounding scheme would yield a polynomial-time $O(1)$-approximation algorithm, contradicting the hardness result of [13].

The second reason is constructive: one can show directly that the standard LP relaxations for certain node-connectivity problems do not have the Jain property; nevertheless, it is possible that the underlying problem may admit a good approximation algorithm via a different LP relaxation or method. Fleischer, Jain, and Williamson [9] studied iterative rounding for minimum-cost node-connectivity problems and showed that it gives a 2-approximation for the *element connectivity* problem – a relaxation of node connectivity in which two "source-sink" paths are permitted to share so-called terminal nodes (but are required to be disjoint on the edges and the Steiner nodes). Their analysis establishes that the Jain property holds for the standard LP relaxation of element connectivity. Crucially, however, their result does not extend to the node-connectivity setting: the requirement function for node connectivity is not weakly supermodular, and the LP does not inherit the Jain property. For the mincost k-node-connected spanning subgraph problem (mincost k-NCSS), which is the core problem of node-connectivity network design, Aazami, Cheriyan, and Laekhanukit [1] gave an

explicit family of bad examples for the standard iterative rounding method. Specifically, they constructed instances for which the setpair LP relaxation has a basic feasible solution x satisfying $\max_e x_e \leq O(1)/\sqrt{k}$, thereby showing that the standard iterative rounding method cannot achieve an approximation ratio better than $\Omega(\sqrt{k})$ for mincost k-NCSS.

Our main result, Theorem 3, differs from the previous results in two essential ways.

(a) Edge-connectivity versus node-connectivity. The results of [13, 9, 1] all concern problems with *node-connectivity* requirements: the requirement function counts internally node-disjoint paths (or a relaxation thereof). Our bad example, by contrast, applies to the Cover Small Cuts problem, which has *edge-connectivity* requirements; recall that the goal is to find a minimum-cost set of links such that every cut of edge-capacity less than λ is covered by at least one link. The standard LP relaxation of Cover Small Cuts (see (LP: Cover Small Cuts)) is a covering LP of the same type as the LP for f-connectivity, yet our construction shows that the Jain property does *not* hold for it. To the best of our knowledge, there is no prior bad example for the Jain property for any problem defined by uniform edge-connectivity requirements.

(b) Hardness versus polyhedral structure. A key distinction between our result and the hardness-based argument of [13] is that Cover Small Cuts *does* admit a polynomial-time $O(1)$-approximation algorithm [6, 3, 15], achieved via the primal-dual method of Williamson et al. [19], rather than iterative rounding. Thus the failure of the Jain property for Cover Small Cuts cannot be attributed to hardness of approximation. This separates Cover Small Cuts from NC-SNDP: for NC-SNDP, the Jain property fails (in part) because the problem is hard to approximate; for Cover Small Cuts, the problem allows an $O(1)$ approximation algorithm, but the known iterative rounding methods via the natural covering LP cannot achieve an $O(1)$ approximation ratio. In the context of network design on undirected graphs with uniform edge-connectivity requirements, our construction indicates that the "limits" of the iterative rounding method are "tighter" than what is indicated by the previous literature.

1.3 Basic notation and definitions

For a positive integer k, we use $[k]$ to denote the set $\{1, 2, \ldots, k\}$. A pair of subsets A, B of V (the ground-set) is said to *cross* if each of the four sets $A \cap B, V - (A \cup B), A - B, B - A$ is nonempty.

For a graph $G = (V, E)$ and $S \subseteq V$, the cut $\delta_E(S)$ refers to the set of edges that have exactly one end-node in S; $\delta_E(S)$ is called a non-trivial cut if $\emptyset \neq S \subsetneq V$.

For each of the minimization problems (in network design or flexible graph connectivity), we use OPT to denote the optimal value (i.e., the minimum cost of an integer solution), and $\mathrm{LP}_{\mathrm{opt}}$ to denote the optimal value of an LP relaxation. The context will resolve potential ambiguities.

2 The Capacitated Graph

Let $k \geq 4$ be an even integer. The capacitated graph $G = (V, E), u$ has $n = 2 + \binom{k}{2}$ nodes, and $m = (n-1) + (k-1) = \binom{k}{2} + k$ edges. The edge capacities are chosen from the set $\{1, 2, 3\}$ and λ (the threshold value for small cuts) is chosen to be 5.

We denote the nodes of G by $v_1, v_2, \ldots, v_n$, as well as by the indices $1, 2, \ldots, n$. For notational convenience, let $s = v_1$ and let $t = v_n$.

The family of small cuts consists of two sub-families. One sub-family corresponds to the node-sets $N_1 = \{v_1\}, N_2 = \{v_1, v_2\}, \ldots, N_{n-1} = \{v_1, v_2, \ldots, v_{n-1}\}$. In other words, for each of the sets $N_i = \{v_1, v_2, \ldots, v_i\}, i = 1, \ldots, n-1$, $\delta_E(N_i)$ is a small cut. We call the sets N_i $(i = 1, \ldots, n-1)$ the *nested sets*. The other sub-family corresponds to the $(k-1)$ node-sets $Q_1 = \{v_2, v_3, \ldots, v_{1+\frac{k}{2}}\}, Q_2 = \{v_{2+\frac{k}{2}}, v_{3+\frac{k}{2}}, \ldots, v_{1+2\cdot\frac{k}{2}}\}, \ldots, Q_{(k-1)} = \{v_{2+(k-2)\cdot\frac{k}{2}}, v_{3+(k-2)\cdot\frac{k}{2}}, \ldots, v_{1+(k-1)\cdot\frac{k}{2}}\}$. Note that these sets form a partition of $V(G)-\{s,t\}$ into $(k-1)$ sets, each of cardinality $k/2$. In other words, for each of the $(k-1)$ "intervals" Q_j $(j = 1, \ldots, k-1)$ formed by picking the node with index $2 + (j-1)\cdot\frac{k}{2}$ and the next $(\frac{k}{2}-1)$ consecutively indexed nodes, $\delta_E(Q_j)$ is a small cut. We call the sets Q_j $(j = 1, \ldots, k-1)$ the *Q-sets*; moreover, by the *first node* of Q_j we mean the node of lowest index in Q_j (i.e., node $2 + (j-1)\cdot\frac{k}{2}$), and by the *last node* of Q_j we mean the node of highest index in Q_j (i.e., node $1 + (j)\cdot\frac{k}{2}$). For notational convenience, let $Q_0 = \{s\} = \{v_1\}$ and let $Q_k = \{t\} = \{v_n\}$; moreover, let $s = v_1$ be the *last node* of Q_0 and let $t = v_n$ be the *first node* of Q_k. (Consequently, in the proofs below, s may only serve as the last node of a Q-set, and t may only serve as a first node. Thus, if we examine the case that a specified node v is a first node, it is guaranteed that $v \neq s$ (i.e., the Q-set containing v is not Q_0); similarly, if we examine the case that a specified node v is a last node, it is guaranteed that $v \neq t$ (i.e., the Q-set containing v is not Q_k)).

For a nested set N_i $(i = 1, \ldots, n-1)$ and a Q-set Q_j $(j = 1, \ldots, k-1)$, observe that $s = v_1 \in N_i - Q_j$ and $t = v_n \notin N_i \cup Q_j$; moreover, if $N_i \cap Q_j$ is nonempty, then either Q_j crosses N_i (i.e., both $Q_j - N_i$ and $Q_j \cap N_i$ are nonempty) or $Q_j \subsetneq N_i$.

There are two types of edges in G. There are $n-1$ edges that form the s,t-path $v_1, v_2, \ldots, v_n$; we use E_1 to denote the edge set of this path. The first edge and the last edge of this path have capacity 2, thus, $u_{v_1v_2} = 2 = u_{v_{n-1}v_n}$. Consider any other edge v_iv_{i+1} $(i = 2, \ldots, n-2)$ of this path. If both v_i and v_{i+1} are in the same Q-set then v_iv_{i+1} has capacity 3, otherwise, v_iv_{i+1} has capacity one. Moreover, there are $(k-1)$ other edges (that are not in the s,t-path), and each of these edges has capacity one; we use E_2 to denote this set of $(k-1)$ edges. The nodes incident to the j-th edge of E_2 $(j = 1, \ldots, k-1)$ have indices $1 + (j-1)\cdot\frac{k}{2}$ and $2 + j\cdot\frac{k}{2}$. Thus, E_2 has an edge between $s = v_1$ and the first node of Q_2, an edge between the last node of Q_{k-2} and $t = v_n$, and, for each $j = 1, \ldots, k-3$, E_2 has an edge between the last node of Q_j and the first node of Q_{j+2}.

The following figure illustrates the construction of the capacitated graph for $k = 4$; the graph has $\binom{k}{2} + 2 = 8$ nodes and $\binom{k}{2} + k = 10$ edges.

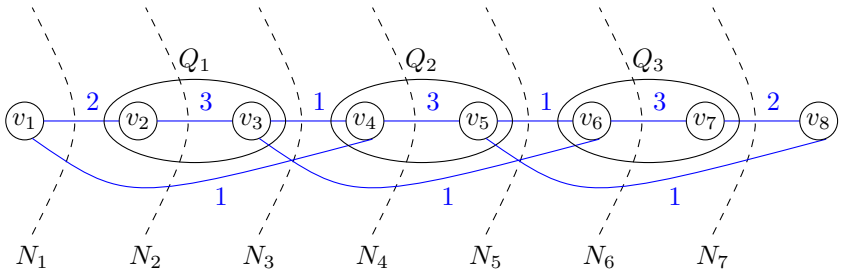


Figure 1 The capacitated graph for $k = 4$. The small cuts are given by the nested sets $N_1, \ldots, N_7$ (indicated by dashed lines) and the sets Q_1, Q_2, Q_3 (indicated by ovals).

We defer the proofs of the next two results to section 5; our proofs are straightforward and use case analysis.

▶ **Proposition 4.** *Each of the nested cuts $\delta(N_i), i = 1, \ldots, n-1$ and each of the Q-cuts $\delta(Q_j), j = 1, \ldots, k-1$ has capacity less than $\lambda = 5$.*

▶ **Proposition 5.** *Every non-trivial cut $\delta(S)$, $\emptyset \subsetneq S \subsetneq V$, of G that is neither a nested cut nor a Q-cut has capacity $\geq \lambda = 5$.*

3 The Links Graph

Let (V, L) denote the graph of the links; thus L is the set of links, and we denote the links by the symbol ℓ, e.g., ℓ_i, ℓ', etc. The number of links is $m = n + k - 2 = \binom{k}{2} + k$. The links are partitioned into k internally node disjoint s,t-paths that we denote by $P_1, \ldots, P_{k-1}, P_k$. The s,t-path P_k has one link $st = v_1 v_n$. Each of the s,t-paths $P_1, \ldots, P_{k-1}$ has $k/2$ internal nodes and has $1 + \frac{k}{2}$ links. Moreover, the sequence of node indices of each of these s,t-paths is an increasing sequence; that is, the node sequence has the form $v_1, v_{i_2}, v_{i_3}, \ldots, v_{i_{(\frac{k}{2}+1)}}, v_n$, where $1 < i_2 < i_3 < \cdots < i_{(\frac{k}{2}+1)} < n$. Each of the nodes $v_2, \ldots, v_{n-1}$ is in one of the s,t-paths $P_1, \ldots, P_{k-1}$; thus, each of these nodes is incident to two links.

Recall that we have $(k-1)$ Q-sets, each with $k/2$ nodes, and $(k-1)$ s,t-paths $P_1, \ldots, P_{k-1}$ each with $k/2$ internal nodes. Each of the s,t-paths P_i $(i = 1, \ldots, k-1)$ is constructed such that it is incident to precisely one node of $k/2$ (of the $(k-1)$) Q-sets, and it is disjoint from the other $(k - 1 - \frac{k}{2})$ Q-sets. The s,t-path P_i $(i = 1, \ldots, k-1)$ is incident to the Q-set with index i and the next $(k/2) - 1$ consecutively indexed Q-sets (with "wrap around"); thus, for $i = 1, \ldots, k-1$ and $j = 1, \ldots, k-1$, the s,t-path P_i and the set Q_j have a node in common iff

$$j \geq i \quad \textbf{and} \quad j \leq \min(i + \frac{k}{2} - 1,\ k-1), \quad \textbf{or}$$
$$j < i \quad \textbf{and} \quad 1 \leq j \leq i - \frac{k}{2}.$$

If an s,t-path P_i and a set Q_j have a node in common, then any one node of Q_j could be assigned to P_i; in other words, our construction does not specify the internal nodes of P_i, it only specifies the Q-sets that intersect P_i.

Let A^{PQ} denote the $(k-1) \times (k-1)$ zero-one matrix that has a row for each s,t-path P_i $(i = 1, \ldots, k-1)$ and a column for each set Q_j $(i = 1, \ldots, k-1)$ such that the entry $A^{PQ}_{i,j}$ is one iff P_i and Q_j have a node in common. Then, by construction, A^{PQ} is a circulant matrix with entries of zero or one such that each row has $k/2$ consecutive ones (with "wrap around") and the remaining $(k - 1 - \frac{k}{2})$ entries of the row are zero. Observe that the number of ones in each row, $k/2$, and the size of the matrix, $k-1$, are relatively prime (i.e., $\gcd(\frac{k}{2}, k-1) = 1$). For an illustration, see the example at the end of section 4.

▶ **Lemma 6.** *The matrix A^{PQ} has rank $(k-1)$. Moreover, $\det(A^{PQ}) = k/2$.*

One can prove this lemma directly, by applying row operations (and applying the matrix determinant lemma [1] to a square matrix with diagonal entries 2 and all other entries one). Alternatively, the lemma follows directly from a result of Hariprasad [11, Theorem 2.1].

Figure 2 illustrates the construction of the links graph for $k = 4$. The graph has $\binom{k}{2} + 2 = 8$ nodes and $\binom{k}{2} + k = 10$ links. The links partition into $k = 4$ (internally node disjoint) s,t-paths $P_1, \ldots, P_4$; the links of $P_1, \ldots, P_4$ are indicated by distinct colours.

[1] https://en.wikipedia.org/wiki/Matrix_determinant_lemma

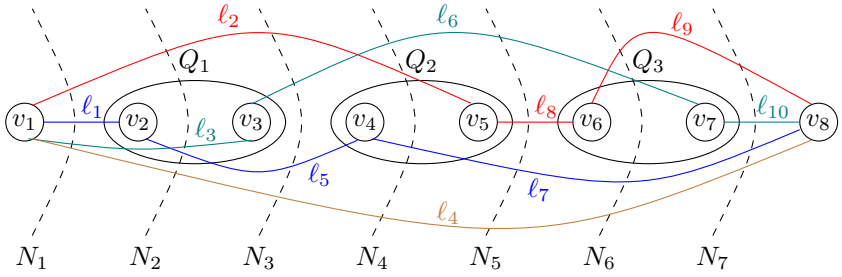


Figure 2 The links graph for $k = 4$. The links partition into s, t-paths $P_1, \ldots, P_4$, and the links of each of these s, t-paths is indicated by a distinct colour. Thus, the links of P_1 are ℓ_1, ℓ_5, ℓ_7, the links of P_2 are ℓ_2, ℓ_8, ℓ_9, the links of P_3 are $\ell_3, \ell_6, \ell_{10}$, and P_4 has the link ℓ_4. The small cuts are given by the nested sets $N_1, \ldots, N_7$ (indicated by dashed lines) and the sets Q_1, Q_2, Q_3 (indicated by ovals).

We construct a basic feasible solution x^* by assigning the value $1/k$ to each link, thus, $x^*_\ell = 1/k, \forall \ell \in L$. Informally speaking, we are assigning a value $1/k$ to each of the s, t-paths $P_1, \ldots, P_k$. It is easy to verify that each of the small cuts, namely, $\delta(N_i)$ $(i = 1, \ldots, n-1)$ and $\delta(Q_j)$ $(j = 1, \ldots, k-1)$, is "tightly covered" by x^*. First, consider a cut $\delta(N_i)$ $(i = 1, \ldots, n-1)$, and observe that each of the s, t-paths $P_1, \ldots, P_k$ has exactly one of its links in $\delta(N_i)$, hence, $x^*(\delta(N_i)) = (k)\frac{1}{k} = 1$. Next, consider a cut $\delta(Q_j)$ $(j = 1, \ldots, k-1)$, and observe that $k/2$ of the s, t-paths $P_1, \ldots, P_k$ have exactly one internal node in Q_j, hence, each of these s, t-paths has exactly two links in $\delta(Q_j)$, therefore, $x^*(\delta(Q_j)) = 2(\frac{k}{2})(\frac{1}{k}) = 1$. Note that the tight constraints at x^* are the covering constraints for the small cuts, and none of the lower bound constraints is tight (since $0 < 1/k < 1$).

The basis matrix A of x^* has one row for each small cut and one column for each link; the entry of A corresponding to a small cut $\delta(S)$ and a link ℓ is one if $\ell \in \delta(S)$, otherwise, the entry is zero. Thus, A is an $m \times m$ matrix with entries of zero or one; recall that $m = n + k - 2 = \binom{k}{2} + k$. In the next section, we show that matrix A has full rank, and that will prove our claim that x^* is a basic feasible solution of the LP relaxation of Cover Small Cuts.

4 The rank of the incidence matrix of small cuts and links

Let A denote the zero-one incidence matrix of the small cuts and the links; A is an $m \times m$ matrix, where $m = n + k - 2 = \binom{k}{2} + k$. The matrix A has rows corresponding to the cuts of the Q-sets followed by rows corresponding to the cuts of the nested sets; the columns of A correspond to the links. In this section, the main goal is to show that $\det(A)$ is non-zero; equivalently, A has full rank. We start by fixing a convenient indexing for the rows and columns of A. The rows $1, \ldots, (k-1)$ of A correspond to the cuts of $Q_1, \ldots, Q_{(k-1)}$, and the rows $k, (k+1), \ldots, m$ of A correspond to the cuts of $N_1, \ldots, N_{(n-1)}$. The links $\ell \in L$ are indexed as follows: The link of the s, t-path P_i $(i = 1, \ldots, k)$ incident to node $s = v_1$ is denoted ℓ_i; thus, each of $\ell_1, \ldots, \ell_k$ is in $\delta(s)$, and there is a bijection between these links and the s, t-path $P_1, \ldots, P_k$. Note that each of the remaining links is incident to one of the nodes $v_2, \ldots, v_{n-1}$, and, for each $i = 2, \ldots, (n-1)$, the unique link that is incident to node v_i and is in the cut $\delta(N_i)$ is assigned the index $(k + i - 1)$. The columns $1, \ldots, m$ of A correspond to the links $\ell_1, \ldots, \ell_m$.

Let us view the matrix A as a 2×2 block matrix $\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$ where A_{11} is the $(k-1) \times (k-1)$ sub-matrix of A whose rows correspond to the cuts of $Q_1, \ldots, Q_{(k-1)}$ and whose columns correspond to the links $\ell_1, \ldots, \ell_{(k-1)}$, and A_{22} is the $(m-k+1) \times (m-k+1)$ sub-matrix of A whose rows correspond to the cuts of $N_1, \ldots, N_{(n-1)}$ and whose columns correspond to the links $\ell_k, \ell_{(k+1)}, \ldots, \ell_m$.

▷ Claim 7. The sub-matrix A_{22} is lower-triangular and all entries on the diagonal are one; thus, $\det(A_{22}) = 1$.

Proof. The link ℓ_k (with end nodes s and t) is in each of the cuts $\delta(N_i)$, $i = 1, \ldots, (n-1)$. For each of the links $\ell_{(k+i-1)}$ where $i = 2, \ldots, (n-1)$, note that the link is in the cut $\delta(N_i)$ and it is in none of the cuts $\delta(N_1), \ldots, \delta(N_{(i-1)})$. ◁

Our plan is to apply row operations to the first $(k-1)$ rows of A in order to transform the submatrix $\begin{pmatrix} A_{11} & A_{12} \end{pmatrix}$ to a submatrix of the form $\begin{pmatrix} (A^{PQ})^\top & \mathbf{0}_{(k-1)\times(m-k+1)} \end{pmatrix}$, where $(A^{PQ})^\top$ denotes the transpose of the matrix A^{PQ} and $\mathbf{0}_{(k-1)\times(m-k+1)}$ denotes a $(k-1) \times (m-k+1)$ matrix of zeros. Therefore, using row operations, we will rewrite the matrix A as the 2×2 block matrix $\begin{pmatrix} (A^{PQ})^\top & \mathbf{0}_{(k-1)\times(m-k+1)} \\ A_{21} & A_{22} \end{pmatrix}$; then, by applying Lemma 6 as well as a result on the determinant of 2×2 block matrices, we see that the determinant of the re-written matrix A is $\det(A^{PQ}) \det(A_{22}) = \det(A^{PQ}) \neq 0$.

See the example at the end of this section for an illustration.

Let us introduce some notation to describe and analyse our row operations. For any set of links L' let $\chi^{L'}$ denote the zero-one row incidence vector of L'; thus, for any link ℓ, $\chi^{L'}_\ell = 1$ iff $\ell \in L'$. Note that rows $1, \ldots, (k-1)$ of A correspond to $\chi^{\delta(Q_1)}, \ldots, \chi^{\delta(Q_{(k-1)})}$. For any link ℓ, let $\phi(\ell)$ denote the index i of the s,t-path P_i that contains ℓ. For each of the links ℓ_i, $i = 1, \ldots, (k-1)$, note that $\phi(\ell_i) = i$, since we have a bijection between these links and the s,t-paths $P_1, \ldots, P_{(k-1)}$. For any set of links L', let $\Phi(L')$ denote the set of indices of the s,t-paths $P_1, \ldots, P_{(k-1)}$ that contain at least one of the links of L'; thus, $\Phi(L') \subseteq \{1, \ldots, k\}$, and index i is in $\Phi(L')$ iff the s,t-path P_i contains one of the links of L'. For example, if L' is a subset of $\{\ell_1, \ldots, \ell_{(k-1)}\}$, then $\Phi(L') = \{i \ : \ \ell_i \in L'\}$.

Consider any row j, $j = 1, \ldots, (k-1)$ of the matrix A; this row-vector is the same as $\chi^{\delta(Q_j)}$. Let g denote the largest index i such that the nested set N_i is disjoint from Q_j, and let h denote the smallest index i such that the nested set N_i contains Q_j. (By our construction, the $k/2$ nodes of Q_j are in the interval $[(g+1), (g+2), \ldots, h]$.)

▷ Claim 8. $\left(\chi^{\delta(Q_j)} - \chi^{\delta(N_h)} + \chi^{\delta(N_g)}\right) = 2\chi^{L'}$, where L' is the set of $k/2$ links in $\delta(Q_j) \cap \delta(N_g)$.

Proof. By construction, every link has its two end nodes in two different Q-sets. Hence, every link $\ell \in \delta(Q_j)$ that has its lower-index node in Q_j is in $\delta(N_h)$; similarly, every link $\ell \in \delta(Q_j)$ that has its higher-index node in Q_j is in $\delta(N_g)$. Moreover, $\delta(N_g) - \delta(Q_j) = \delta(N_h) - \delta(Q_j)$ is the set of links that have their lower-index node in N_g and their higher-index node in $V - N_h$. The equation in the claim follows from these statements. ◁

In the above claim, note that L' is contained in the cut of a Q-set, hence, the link $\ell_k = st$ cannot be in L'.

Our plan is to (separately) apply the row operations described in the above claim to each of the rows $j = 1, \ldots, (k-1)$ of A. The next lemma explains how to apply row operations to $\chi^{L'}$, where L' is any set of $k/2$ links contained in some nested cut $\delta(N_i)$ and $\ell_k \notin L'$, to transform $\chi^{L'}$ to the vector $\chi^{\widehat{L}}$, where $\widehat{L} = \{\ell_i \ : \ i \in \Phi(L')\}$. Informally speaking, we can

apply row operations to "map" any row vector $\chi^{L'}$, where $L' \subset \delta(N_i) - \{\ell_k\}$ and $|L'| = k/2$, to the row incidence vector (padded with zeros on the right) of the $k/2$ s,t-paths among $P_1, \ldots, P_{(k-1)}$ that contain (at least) one of the links of L'. Thus, by applying the above claim and the next lemma, we "map" each row of the submatrix $\begin{pmatrix} A_{11} & A_{12} \end{pmatrix}$ to a distinct row of the submatrix $\begin{pmatrix} (A^{PQ})^\top & \mathbf{0}_{(k-1)\times(m-k+1)} \end{pmatrix}$.

▶ **Lemma 9.** *Let $i = 1, \ldots, n-1$, and let $L' \subset \delta(N_i) - \{\ell_k\}$ be any set of $k/2$ links. By applying row operations, one can transform the vector $\chi^{L'}$ to the vector $\chi^{\widehat{L}}$, where $\widehat{L} = \{\ell_i \,:\, i \in \Phi(L')\}$.*

Proof. By induction on the index of the nested set whose cut contains L'.

Let h be the smallest index such that the nested cut $\delta(N_h)$ contains L'. If $h = 1$, then we have $\widehat{L} = L'$ and the lemma is proved.

Now, suppose $h \geq 2$; thus, L' contains a link ℓ_j with $j > k$. Observe that L' contains the link $\ell_{(h+k-1)}$ which is incident to node v_h and is in the cut $\delta(N_h)$; informally speaking, the presence of $\ell_{(h+k-1)}$ in L' certifies that L' is contained in $\delta(N_h)$ and L' is not a subset of $\delta(N_j)$ for any $j < h$. Let $\overline{L}$ denote the set of links $\delta(N_h) - L'$; $\overline{L}$ is a "complementary set" of $k/2$ links w.r.t. $\delta(N_h)$ and L'. For every link $\ell_j \in \overline{L}$, observe that $j < (h+k-1)$ (since $\ell_j \in \delta(N_h)$ and $\ell_{(h+k-1)} \notin \overline{L}$).

Let g be the smallest index such that the nested cut $\delta(N_g)$ contains $\overline{L}$. Again, observe that $\overline{L}$ contains the link $\ell_{(g+k-1)}$ which is incident to node v_g and is in the cut $\delta(N_g)$. Clearly, $g < h$ and $\delta(N_g) \neq \delta(N_h)$, since $\ell_{(h+k-1)} \in \delta(N_h)$ and $\ell_{(h+k-1)} \notin \delta(N_g)$.

▷ Claim 10. There exists a set of $k/2$ links of $\delta(N_g) - \{\ell_k\}$, call it L'', such that
(i) $\chi^{L''} = \chi^{L'} - \chi^{\delta(N_h)} + \chi^{\delta(N_g)}$, and
(ii) $\Phi(L'') = \Phi(L')$.

Proof. Observe that $\overline{L} = \delta(N_h) - L'$, and that $\overline{L}$ is contained in $\delta(N_g)$; also, note that ℓ_k is in $\overline{L}$. Let $L'' = \delta(N_g) - \overline{L}$; observe that L'' is a set of $k/2$ links and $\ell_k \notin L''$. Hence, we have $\chi^{\delta(N_g)} - \chi^{\delta(N_h)} = \chi^{L''} - \chi^{L'}$. This proves part (i) of the claim.

Moreover, note that each of the nested cuts contains exactly one link from each of the s,t-paths $P_1, \ldots, P_k$; that is, for each $i = 1, \ldots, n-1$, $\Phi(\delta(N_i)) = \{1, \ldots, k\}$. Hence, $\Phi(L') = \Phi(\delta(N_h) - \overline{L}) = \{1, \ldots, k\} - \Phi(\overline{L}) = \Phi(\delta(N_g) - \overline{L}) = \Phi(L'')$. This proves part (ii) of the claim. ◁

In the induction step (of the proof of the lemma), we apply the above claim to replace the set L' by the set L''; note that L'' has the properties of L' (listed in the lemma statement) and $\Phi(L'') = \Phi(L')$. Moreover, L'' is contained in the cut of a nested set of smaller index than the index of any nested set whose cut contains L'.

Thus, the lemma follows by induction. ◀

Our main result of this section follows from the above discussion.

▶ **Theorem 3.** *Let $k \geq 4$ be an even integer. There is an instance of the* Cover Small Cuts *problem $G = (V, E), L, u$ such that the LP relaxation* (LP: Cover Small Cuts) *has a basic feasible solution x^* such that every positive variable of x^* has value $1/k$.*

4.1 Example with $k = 4$

In this subsection, we fix $k = 4$, and we illustrate our row operations, in particular, the row operations of Claims 8,10, on the links graph shown in Figure 2. The links partition into s,t-paths $P_1, \ldots, P_4$, and the links of P_1 are ℓ_1, ℓ_5, ℓ_7, the links of P_2 are ℓ_2, ℓ_8, ℓ_9, the links of P_3 are $\ell_3, \ell_6, \ell_{10}$, and P_4 has the link ℓ_4.

The incidence matrix A^{PQ} of the s,t-paths P_1, P_2, P_3 and the Q-sets is given below.

$$\begin{array}{ccc} Q_1 & Q_2 & Q_3 \\ \left(\begin{array}{ccc} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{array}\right) & & \begin{array}{c} P_1 \\ P_2 \\ P_3 \end{array} \end{array}$$

The incidence matrix A of the small cuts and the links is given below. Note that the submatrix A_{22} (of the nested cuts and the links $\ell_4, \ldots, \ell_{10}$) is lower-triangular with diagonal entries of one.

$$\begin{array}{ccc|cccccccc} \ell_1 & \ell_2 & \ell_3 & \ell_4 & \ell_5 & \ell_6 & \ell_7 & \ell_8 & \ell_9 & \ell_{10} & \\ \hline 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & Q_1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & Q_2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & Q_3 \\ \hline 1 & 1 & 1 & \textcircled{1} & 0 & 0 & 0 & 0 & 0 & 0 & N_1 \\ 0 & 1 & 1 & 1 & \textcircled{1} & 0 & 0 & 0 & 0 & 0 & N_2 \\ 0 & 1 & 0 & 1 & 1 & \textcircled{1} & 0 & 0 & 0 & 0 & N_3 \\ 0 & 1 & 0 & 1 & 0 & 1 & \textcircled{1} & 0 & 0 & 0 & N_4 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & \textcircled{1} & 0 & 0 & N_5 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & \textcircled{1} & 0 & N_6 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & \textcircled{1} & N_7 \end{array}$$

We describe the row operations for $\delta(Q_1), \delta(Q_2), \delta(Q_3)$, i.e., rows 1,2,3 of A; in particular, we describe the row operations of Claims 8,10.

Q_1: $\chi^{\delta(Q_1)} = \chi^{\{\ell_1,\ell_3,\ell_5,\ell_6\}}$. In Claim 8, for $j = 1$, we have $h = 3, g = 1$. $\left(\chi^{\delta(Q_1)} - \chi^{\delta(N_3)} + \chi^{\delta(N_1)}\right) = 2\chi^{\{\ell_1,\ell_3\}}$.
Note that $\Phi(\delta(Q_1)) = \{1, 3\}$.

Q_2: $\chi^{\delta(Q_2)} = \chi^{\{\ell_2,\ell_5,\ell_7,\ell_8\}}$. In Claim 8, for $j = 2$, we have $h = 5, g = 3$. $\left(\chi^{\delta(Q_2)} - \chi^{\delta(N_5)} + \chi^{\delta(N_3)}\right) = 2\chi^{\{\ell_2,\ell_5\}}$.
Next, Claim 10 is applied with $L' = \{\ell_2, \ell_5\}$, and we have $h = 2, g = 1$. $\chi^{\{\ell_2,\ell_5\}} - \chi^{\delta(N_2)} + \chi^{\delta(N_1)} = \chi^{\{\ell_1,\ell_2\}}$.
Note that $\Phi(\delta(Q_2)) = \{1, 2\}$.

Q_3: $\chi^{\delta(Q_3)} = \chi^{\{\ell_6,\ell_8,\ell_9,\ell_{10}\}}$. In Claim 8, for $j = 3$, we have $h = 7, g = 5$. $\left(\chi^{\delta(Q_3)} - \chi^{\delta(N_7)} + \chi^{\delta(N_5)}\right) = 2\chi^{\{\ell_6,\ell_8\}}$.
Next, Claim 10 is applied with $L' = \{\ell_6, \ell_8\}$, and we have $h = 5, g = 4$. $\chi^{\{\ell_6,\ell_8\}} - \chi^{\delta(N_5)} + \chi^{\delta(N_4)} = \chi^{\{\ell_2,\ell_6\}}$.
Next, Claim 10 is applied with $L' = \{\ell_2, \ell_6\}$, and we have $h = 3, g = 2$. $\chi^{\{\ell_2,\ell_6\}} - \chi^{\delta(N_3)} + \chi^{\delta(N_2)} = \chi^{\{\ell_2,\ell_3\}}$.
Note that $\Phi(\delta(Q_3)) = \{2, 3\}$.

5 The Capacitated Graph: Deferred Proofs

In this section, we present the proofs of Propositions 4 and 5, deferred from section 2.

▶ **Proposition 4.** *Each of the nested cuts $\delta(N_i), i = 1, \ldots, n-1$ and each of the Q-cuts $\delta(Q_j), j = 1, \ldots, k-1$ has capacity less than $\lambda = 5$.*

Proof. We examine a few cases.

Nested cut $\delta(N_i)$ such that N_i does not cross any Q-set: Then, either $2 \leq i \leq n-2$ and there is a largest index $j = 1, \ldots, k-2$ such that Q_j is contained in N_i, or else $i \in \{1,\ (n-1)\}$.

If $i = 1$, then $\delta(N_i)$ has two edges, namely, (i) the E_1-edge v_1v_2 of capacity 2, and (ii) the E_2-edge (of capacity one) between $s = v_1$ and the first node of Q_2; clearly, $\delta(N_1)$ has capacity $3 < \lambda = 5$. Similarly, if $i = (n-1)$, then $\delta(N_i)$ has two edges and has capacity $3 < \lambda = 5$.

Otherwise, $\delta(N_i)$ has three edges, namely, (i) the E_1-edge v_iv_{i+1} of capacity one, (ii) the E_2-edge (of capacity one) between the last node of Q_{j-1} and the first node of Q_{j+1}, and (iii) the E_2-edge (of capacity one) between the last node of Q_j and the first node of Q_{j+2}, Thus, the cut $\delta(N_i)$ has capacity $3 < \lambda = 5$.

Nested cut $\delta(N_i)$ such that N_i crosses a Q-set: Suppose that N_i crosses Q_j, where $j = 1, \ldots, k-1$. Clearly, the first node of Q_j is in N_i and the last node of Q_j is not in N_i. Note that $\delta(N_i)$ has two edges, namely, the E_1-edge v_iv_{i+1} of capacity 3, and the E_2-edge (of capacity one) between the last node of Q_{j-1} and the first node of Q_{j+1}. Thus, the cut $\delta(N_i)$ has capacity $4 < \lambda = 5$.

Q-cut $\delta(Q_j)$: First, suppose that $j = 2, \ldots, (k-2)$. Observe that $\delta(Q_j)$ consists of the following four unit-capacity edges. There are two edges of E_1 in the cut, namely, the edge between the last node of Q_{j-1} and the first node of Q_j, and the edge between the last node of Q_j and the first node of Q_{j+1}. Moreover, there are two edges of E_2 in the cut; namely, the edge between the last node of Q_{j-2} and the first node of Q_j, and the edge between the last node of Q_j and the first node of Q_{j+2}.

The cuts $\delta(Q_1)$ and $\delta(Q_{k-1})$ have capacity four since each of these cuts has only one edge of E_2 (since no E_2-edge is incident to the first node of Q_1 or to the last node of Q_{k-1}), and has two edges of E_1, one edge of capacity 2 (i.e., $sv_2 \in \delta(Q_1)$ and $v_{n-1}t \in \delta(Q_{k-1})$) and the other edge of capacity one. ◀

▶ **Proposition 5.** *Every non-trivial cut $\delta(S)$, $\emptyset \subsetneq S \subsetneq V$, of G that is neither a nested cut nor a Q-cut has capacity $\geq \lambda = 5$.*

Proof. We denote the nodes $v_1, \ldots, v_n$ by their indices $1, \ldots, n$ for this proof. (Notational remark: In this proof, the symbols ℓ_1, ℓ_2, etc., refer to the indices $1, \ldots, n$ rather than to links in L; in fact, links are not relevant in this section.) W.l.o.g., assume that S is a set of nodes such that $1 \in S$ (thus, S contains v_1), $S \neq N_i, i = 1, \ldots, n-1$, and $S \neq (V - Q_j), j = 1, \ldots, k-1$. Let $\ell_1 = 1$ and let r_1 denote the smallest index such that $r_1 \in S$ and $(r_1 + 1) \notin S$; note that r_1 is well defined, since $S \neq V$; possibly, $r_1 = \ell_1$. Thus, $[\ell_1, \ldots, r_1]$ is a maximal interval contained in S. Clearly, $S \neq [\ell_1, \ldots, r_1]$, otherwise, we would have $S = N_{r_1}$ (contradicting our assumption that S is not a nested set). Thus, $r_1 \leq (n-2)$. Let ℓ_2 be the smallest index $> r_1$ that is in S; thus, $\ell_2 = \min_i\{i \in S, i \in [(r_1+1), \ldots, n]\}$. Moreover, let r_2 be the smallest index $\geq \ell_2$ such that either $r_2 = n$ or else $(r_2 + 1) \notin S$; thus, $r_2 = n$ if $[\ell_2, \ldots, n] \subset S$, and $r_2 = \min\{i \in S, i+1 \notin S, i \in [\ell_2, \ldots, n]\}$ if $[\ell_2, \ldots, n] \not\subset S$. Then, $[\ell_2, \ldots, r_2]$ is another maximal interval contained in S. Clearly, both the E_1-edges $v_{r_1}v_{(r_1+1)}$ and $v_{(\ell_2-1)}v_{\ell_2}$ are in $\delta(S)$.

For any node i $(i = 1, \ldots, n)$, we use $q(i)$ to denote the index of the (unique) Q-set that contains i; thus, $q(i) = \lceil (i-1)/\frac{k}{2} \rceil$.

We examine a few cases and show that $\delta(S)$ has capacity $\geq \lambda = 5$ in each case.

- Node r_1 is not the last node of its Q-set, $Q_{q(r_1)}$, and node ℓ_2 is not the first node of its Q-set, $Q_{q(\ell_2)}$:
  Note that $r_1 \neq s = 1$ and $\ell_2 \neq t = n$.

Then the E_1-edge $v_{r_1}v_{(r_1+1)}$ has capacity 3, and the E_1-edge $v_{(\ell_2-1)}v_{\ell_2}$ has capacity 3, hence, $\delta(S)$ has capacity $\geq 6 > \lambda = 5$.

- Node r_1 is not the last node of its Q-set, $Q_{q(r_1)}$, and node ℓ_2 is the first node of its Q-set, $Q_{q(\ell_2)}$:
  Note that $r_1 \neq s = 1$, and, possibly, $\ell_2 = t = n$.
  Thus, $r_1 \geq 2$. Then the E_1-edge $v_{r_1}v_{(r_1+1)}$ has capacity 3.
  Suppose $\ell_2 = t = n$. Then the E_1-edge $v_{(\ell_2-1)}v_{\ell_2}$ has capacity two. Thus, $\delta(S)$ has capacity $\geq 5 = \lambda = 5$.
  Now, suppose $\ell_2 \neq t = n$. Then the E_1-edge $v_{(\ell_2-1)}v_{\ell_2}$ has capacity one. Note that $q(r_1) \geq 1$ and $2 \leq q(\ell_2) \leq (k-1)$. We have two subcases, depending on $q(\ell_2) - q(r_1)$.
  $q(\ell_2) = q(r_1) + 1$: Then we have two further subcases.
  - **(i)** First, suppose that $r_2 = n$; thus, $[\ell_2, \ldots, n]$ is a maximal interval contained in S. Then the last node of $Q_{q(r_1)}$ is not in S and the first node of $Q_{(q(r_1)+2)}$ is in S. Then the E_2-edge between the last node of $Q_{q(r_1)}$ and the first node $Q_{(q(r_1)+2)}$ is in $\delta(S)$. Thus, the cut $\delta(S)$ has capacity $\geq 5 = \lambda = 5$.
  - **(ii)** Now, suppose that $r_2 < n$. Then the E_1-edge $v_{r_2}v_{(r_2+1)}$ is in $\delta(S)$, and it has capacity one or more. Thus, the cut $\delta(S)$ has capacity $\geq 5 = \lambda = 5$.

  $q(\ell_2) \geq q(r_1) + 2$: Then the last node of $Q_{(q(r_1)-1)}$ is in S and the first node of $Q_{(q(r_1)+1)}$ is not in S. Then the E_2-edge between the last node of $Q_{(q(r_1)-1)}$ and the first node $Q_{(q(r_1)+1)}$ is in $\delta(S)$. Thus, the cut $\delta(S)$ has capacity $\geq 5 = \lambda = 5$.
- Node r_1 is the last node of its Q-set, $Q_{q(r_1)}$, and node ℓ_2 is not the first node of its Q-set, $Q_{q(\ell_2)}$:
  Possibly, $r_1 = s = 1$, and recall that $r_1 \leq (n-2)$. Note that $\ell_2 \neq t = n$.
  First, suppose $r_1 = 1$. Then the E_1-edge $v_{r_1}v_{(r_1+1)}$ has capacity two, and the E_1-edge $v_{(\ell_2-1)}v_{\ell_2}$ has capacity 3. Hence, $\delta(S)$ has capacity $\geq 5 = \lambda = 5$.
  Now, suppose $r_1 \geq 2$. Then, $1 \leq q(r_1) \leq (k-2)$. Then the E_1-edge $v_{r_1}v_{(r_1+1)}$ has capacity one, and the E_1-edge $v_{(\ell_2-1)}v_{\ell_2}$ has capacity 3. Clearly, $\delta(S)$ has both these edges, as well as the E_2-edge between the last node of $Q_{q(r_1)-1}$ and the node (r_1+1) (first node of $Q_{q(r_1)+1}$). Hence, $\delta(S)$ has capacity $\geq 5 = \lambda = 5$.
- Node r_1 is the last node of its Q-set, $Q_{q(r_1)}$, and node ℓ_2 is the first node of its Q-set, $Q_{q(\ell_2)}$: Possibly, $r_1 = s = 1$, and recall that $r_1 \leq (n-2)$. Possibly, $\ell_2 = t = n$.
  Note that $q(r_1) \geq 0$ and $2 \leq q(\ell_2) \leq (k)$.
  The E_1-edge $v_{(\ell_2-1)}v_{\ell_2}$ has capacity one or two.
  If $r_1 = s$, then the E_1-edge $v_{r_1}v_{(r_1+1)}$ has capacity two, otherwise, this edge has capacity one.
  If $r_1 \neq s$, then $q(r_1) \geq 1$; moreover, $(r_1+1) \notin S$ and (r_1+1) is the first node of its Q-set, $Q_{q(r_1+1)}$; hence, the E_2-edge between the last node of $Q_{(q(r_1)-1)}$ and (r_1+1) is in $\delta(S)$.
  At this point in the analysis, it is clear that $\delta(S)$ has capacity at least 3.
  We have two subcases, depending on $q(\ell_2) - q(r_1)$.
  $q(\ell_2) \geq q(r_1) + 3$: Then the interval $[(r_1+1), \ldots, (\ell_2-1)]$ (which is a subset of $V - S$) contains two or more Q-sets. We will show that the second and the last-but-one of the Q-sets each contribute an E_2-edge to $\delta(S)$.
  Since r_1 is the last node of its Q-set, $r_1 \in S$, and the first node of $Q_{(q(r_1)+2)}$ is not in S, the E_2-edge between these two nodes is in $\delta(S)$.
  Similarly, since ℓ_2 is the first node of its Q-set, $\ell_2 \in S$, and the last node of $Q_{(q(\ell_2)-2)}$ is not in S, the E_2-edge between these two nodes is in $\delta(S)$.
  Thus, $\delta(S)$ has capacity $\geq 5 = \lambda = 5$.

$q(\ell_2) \leq q(r_1) + 2$: In this case, $q(\ell_2) = q(r_1) + 2$, since the first node of $Q_{(q(r_1)+1)}$, which is $(r_1 + 1)$, is not in S, and ℓ_2 (which is the first node of its Q-set) is in S.
Observe that $r_2 \neq n$; that is, $r_2 \leq (n-1)$ and the node (r_2+1) is not in S. Otherwise, we would have $S = V - Q_{(q(r_1)+1)}$, contradicting our assumption that S is not the complement of a Q-set.
First, suppose $r_2 = (n-1)$. Then $\delta(S)$ has the edge $v_{(n-1)}v_n$ of capacity two. Hence, overall, $\delta(S)$ has capacity $\geq 5 = \lambda = 5$.
Now, suppose $r_2 \leq (n-2)$. If the E_1-edge $v_{r_2}v_{(r_2+1)}$ has capacity 3, then, overall, $\delta(S)$ has capacity $\geq 6 = \lambda = 5$. Otherwise, the E_1-edge $v_{r_2}v_{(r_2+1)}$ has capacity one. Observe that the node r_2 is the last node of its Q-set; moreover, the node (r_2+1) is the first node of its Q-set and $q(r_2+1) \leq (k-1)$. If the interval $[(r_2+1), (r_2+2), \ldots, n]$ has one or more nodes of S, then $\delta(S)$ has one more E_1-edge (besides $v_{r_2}v_{(r_2+1)}$) of capacity one or more, hence, overall, $\delta(S)$ has capacity $\geq 5 = \lambda = 5$. Finally, suppose that the interval $[(r_2+1), (r_2+2), \ldots, n]$ has no nodes of S. Then, the node r_2 is in S and the first node of $Q_{(q(r_2)+2)}$ is not in S, hence, the E_2 edge between these two nodes is in $\delta(S)$. Thus, overall, $\delta(S)$ has capacity $\geq 5 = \lambda = 5$. ◀

References

1 Ashkan Aazami, Joseph Cheriyan, and Bundit Laekhanukit. A bad example for the iterative rounding method for mincost k-connected spanning subgraphs. *Discret. Optim.*, 10(1):25–41, 2013. doi:10.1016/j.disopt.2012.10.002.

2 Ishan Bansal. A Global Analysis of the Primal-Dual Method for Pliable Families. *CoRR*, abs/2308.15714, 2024. doi:10.48550/arXiv.2308.15714.

3 Ishan Bansal. A global analysis of the primal-dual method for edge augmentation problems. In Nicole Megow and Amitabh Basu, editors, *Integer Programming and Combinatorial Optimization - 26th International Conference, IPCO 2025, Baltimore, MD, USA, June 11-13, 2025, Proceedings*, Lecture Notes in Computer Science, pages 58–71. Springer, 2025. doi:10.1007/978-3-031-93112-3_5.

4 Ishan Bansal, Joe Cheriyan, Sanjeev Khanna, and Miles Simmons. Improved approximation algorithms for capacitated network design and flexible graph connectivity. In Keren Censor-Hillel, Fabrizio Grandoni, Joël Ouaknine, and Gabriele Puppis, editors, *52nd International Colloquium on Automata, Languages, and Programming, ICALP 2025, Aarhus, Denmark, July 8-11, 2025*, LIPIcs, pages 20:1–20:20, n.p., 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2025.20.

5 Ishan Bansal, Joseph Cheriyan, Logan Grout, and Sharat Ibrahimpur. Improved Approximation Algorithms by Generalizing the Primal-Dual Method Beyond Uncrossable Functions. In *Proceedings of the 50th International Colloquium on Automata, Languages, and Programming*, volume 261, article 15, pages 1–19, 2023. doi:10.4230/LIPIcs.ICALP.2023.15.

6 Ishan Bansal, Joseph Cheriyan, Logan Grout, and Sharat Ibrahimpur. Improved approximation algorithms by generalizing the primal-dual method beyond uncrossable functions. *Algorithmica*, 86(8):2575–2604, 2024. doi:10.1007/s00453-024-01235-2.

7 Sylvia C. Boyd, Joseph Cheriyan, Arash Haddadan, and Sharat Ibrahimpur. Approximation algorithms for flexible graph connectivity. *Math. Program.*, 204(1):493–516, 2024. doi:10.1007/s10107-023-01961-5.

8 Deeparnab Chakrabarty, Chandra Chekuri, Sanjeev Khanna, and Nitish Korula. Approximability of Capacitated Network Design. *Algorithmica*, 72(2):493–514, 2015. doi:10.1007/s00453-013-9862-4.

9 Lisa Fleischer, Kamal Jain, and David P. Williamson. Iterative rounding 2-approximation algorithms for minimum-cost vertex connectivity problems. *J. Comput. Syst. Sci.*, 72(5):838–867, 2006. doi:10.1016/j.jcss.2005.05.006.

10 Michel X. Goemans, Andrew V. Goldberg, Serge A. Plotkin, David B. Shmoys, Éva Tardos, and David P. Williamson. Improved Approximation Algorithms for Network Design Problems. In *Proceedings of the 5th Symposium on Discrete Algorithms*, pages 223–232, 1994. URL: http://dl.acm.org/citation.cfm?id=314464.314497.

11 M. Hariprasad. Determinant of binary circulant matrices. *Special Matrices*, 7(1):92–94, 2019. doi:10.1515/spma-2019-0008.

12 Kamal Jain. A Factor 2 Approximation Algorithm for the Generalized Steiner Network Problem. *Combinatorica*, 21(1):39–60, 2001. doi:10.1007/s004930170004.

13 Guy Kortsarz, Robert Krauthgamer, and James R. Lee. Hardness of approximation for vertex-connectivity network design problems. *SIAM J. Comput.*, 33(3):704–720, 2004. doi:10.1137/S0097539702416736.

14 Lap Chi Lau, R. Ravi, and Mohit Singh. *Iterative Methods in Combinatorial Optimization*. Cambridge Texts in Applied Mathematics. Cambridge University Press, Cambridge, UK, 2011. doi:10.1017/CBO9780511977152.

15 Zeev Nutov. Tight Analysis of the Primal-Dual Method for Edge-Covering Pliable Set Families. In Pawel Gawrychowski, Filip Mazowiecki, and Michal Skrzypczak, editors, *50th International Symposium on Mathematical Foundations of Computer Science, MFCS 2025, Warsaw, Poland, August 25-29, 2025*, LIPIcs, pages 82:1–82:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.MFCS.2025.82.

16 Zeev Nutov. A tight example for approximation ratio 5 for covering small cuts by the primal-dual method. *CoRR*, abs/2512.08350, 2025. doi:10.48550/arXiv.2512.08350.

17 Miles Simmons. Cover Small Cuts and Flexible Graph Connectivity Problems. Master's thesis, University of Waterloo, Waterloo, ON, Canada, September 2025. URL: https://hdl.handle.net/10012/22426.

18 Miles Simmons, Ishan Bansal, and Joe Cheriyan. A 5-Approximation Analysis for the Cover Small Cuts Problem. *CoRR*, abs/2602.01462, 2026. doi:10.48550/arXiv.2602.01462.

19 David P. Williamson, Michel X. Goemans, Milena Mihail, and Vijay V. Vazirani. A Primal-Dual Approximation Algorithm for Generalized Steiner Network Problems. *Combinatorica*, 15(3):435–454, 1995. doi:10.1007/BF01299747.

Hardness of the Binary Covering Radius Problem in Large ℓ_p Norms

Huck Bennett
University of Colorado Boulder, CO, USA

Peter Ly
University of Colorado Boulder, CO, USA

Abstract

We study the hardness of the γ-approximate decisional Covering Radius Problem on lattices in the ℓ_p norm (γ-GapCRP$_p$). Specifically, we prove that there is an explicit function $\gamma(p)$, with $\gamma(p) > 1$ for $p > p_0 \approx 35.31$ and $\lim_{p\to\infty} \gamma(p) = 9/8$, such that for any constant $\varepsilon > 0$, $(\gamma(p) - \varepsilon)$-GapCRP$_p$ is NP-hard. This shows the first hardness of GapCRP$_p$ for explicit $p < \infty$. Work of Haviv and Regev (CCC, 2006 and CJTCS, 2012) previously showed Π_2-hardness of approximation for GapCRP$_p$ for all sufficiently large (but non-explicit) finite p and for $p = \infty$.

In fact, our hardness results hold for a variant of GapCRP called the *Binary Covering Radius Problem* (BinGapCRP). The Binary Covering Radius Problem trivially reduces to both GapCRP and the decisional Linear Discrepancy Problem (LinDisc) in any norm in an approximation-preserving way. We also show Π_2-hardness of $(9/8 - \varepsilon)$-BinGapCRP in the ℓ_∞ norm for any constant $\varepsilon > 0$.

Our work extends and heavily uses the work of Manurangsi (IPL, 2021), which showed Π_2-hardness of $(9/8 - \varepsilon)$-LinDisc in the ℓ_∞ norm.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases Covering radius problem, linear discrepancy, hardness of approximation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.10

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2603.03219

Funding Supported by NSF Award No. 2432132.

Acknowledgements We thank Ishay Haviv, Pasin Manurangsi, Daniele Micciancio, and Noah Stephens-Davidowitz for helpful comments and references.

1 Introduction

The *covering radius* $\mu(\mathcal{L})$ of a lattice $\mathcal{L}$ is the maximum distance of a point in the span of $\mathcal{L}$ to a vector in $\mathcal{L}$. It is one of the most fundamental lattice quantities. Accordingly, the (decisional) Covering Radius Problem ((Gap)CRP) is among the most fundamental lattice problems. Indeed, it is connected to the main applications of lattices in computer science. Approximate GapCRP is one of the worst-case problems from which there is a reduction to the average-case problems used in lattice-based cryptography [32], and it is related to fast algorithms for integer programming (see, e.g., [8]).


© Huck Bennett and Peter Ly;
licensed under Creative Commons License CC-BY 4.0

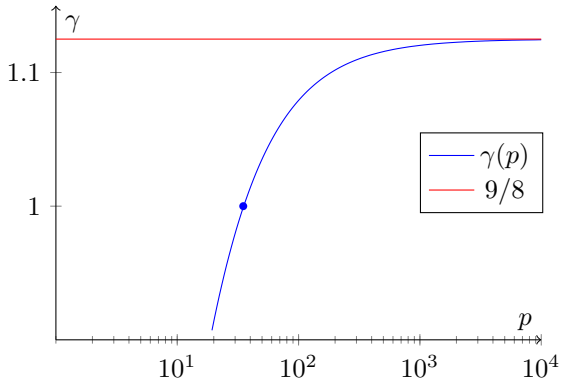


Figure 1 A linear-log plot of the approximation factor $\gamma = \gamma(p)$ for which we show NP-hardness of $(\gamma - \varepsilon)$-BinGapCRP in the ℓ_p norm (in blue; see Theorem 1). The function $\gamma(p)$, specified in Equation (1), is greater than 1 for all $p > p_0 = 35.310188...$ and satisfies $\lim_{p\to\infty} \gamma(p) = 9/8$. (The blue dot is at the point $(p, \gamma) = (p_0, 1)$, where p_0 is the unique value of p such that $\gamma(p) = 1$. The horizontal asymptote of 9/8 is shown in red.) This factor of 9/8 also appears in our Π_2-hardness result for BinGapCRP$_\infty$ (Theorem 2) and in the work of Manurangsi [29] on LinDisc, which is the foundation of our work..

Yet, the hardness of GapCRP is perhaps the least well understood among major lattice problems. In particular, it is not even known that *exact* GapCRP in the ℓ_2 norm is NP-hard, even though it is plausibly NP-hard to approximate within any constant $\gamma \geq 1$ and Π_2-hard to approximate to within a small constant $\gamma > 1$. This stands in contrast to, e.g., the decisional Shortest Vector Problem (γ-GapSVP), which a long line of elegant work [2, 31, 25, 24, 36, 21, 33] showed to be NP-hard in any ℓ_p norm for any constant $\gamma \geq 1$.[1]

Seminal work of Haviv and Regev [20] was the first – and prior to this work, only – to study the hardness of GapCRP. (Other work has studied the complexity of GapCRP more generally, including limitations on its hardness [15, 28]. See Section 1.2.) Let γ-GapCRP$_p$ be γ-approximate GapCRP in the ℓ_p norm. Haviv and Regev [20] showed that for any constant $\varepsilon > 0$, $(3/2 - \varepsilon)$-GapCRP$_\infty$ is Π_2-hard, and that for all sufficiently large p there exists $\gamma = \gamma(p) > 1$ such that γ-GapCRP$_p$ is Π_2-hard. However, the cutoff for "sufficiently large" in their work is non-explicit and presumably quite high, and so in some sense not much more is known about the hardness of GapCRP$_p$ for any explicit, finite p than for $p = 2$.

1.1 Our Results

In this work, we address the problem above by showing NP-hardness of $(\gamma - \varepsilon)$-GapCRP$_p$ for all p greater than an *explicit* value $p_0 \approx 35.31$ with an explicit approximation factor $\gamma = \gamma(p) > 1$. This is the first new hardness result for GapCRP since [20], which was first published 20 years ago.

In fact, we show hardness for a new promise variant of γ-GapCRP$_p$ called γ-BinGapCRP$_p$ where in the YES case every vector in the fundamental parallelepiped $\mathcal{P}(\mathbf{B}) := \mathbf{B} \cdot [0,1]^n$ of the input basis $\mathbf{B} \in \mathbb{R}^{m\times n}$ is close in ℓ_p distance to a lattice vector $\mathbf{Bx}$ with a *binary* coefficient vector $\boldsymbol{x} \in \{0,1\}^n$. That is, in the YES case, for every $\boldsymbol{w} \in [0,1]^n$ there exists $\boldsymbol{x} \in \{0,1\}^n$ such that $\|\mathbf{B}(\boldsymbol{w} - \boldsymbol{x})\|_p$ is small. This implies that the covering radius $\mu_p(\mathcal{L}(\mathbf{B}))$

[1] Formally, this NP-hardness is only known under *randomized reductions*, although several works – some of which are very recent [16, 22, 17, 38]! – have derandomized the reductions in some cases.

is small. (See Equation (4) and the surrounding text for a formal definition of the covering radius and a discussion of why this is true.) In the NO case, there exists a vector in $\mathcal{P}(\mathbf{B})$ that is far from *every* lattice vector $\mathbf{B}\boldsymbol{x}$ for $\boldsymbol{x} \in \mathbb{Z}^n$, as with standard γ-GapCRP$_p$.

BinGapCRP is reminiscent of the "binary coefficient vector" variant of the Closest Vector Problem often used in hardness reductions (see, e.g., [13, Definition 7] and [3, Definition 3.1]), and may be of independent interest. It has the YES instances of the Linear Discrepancy Problem (LinDisc) and the NO instances of GapCRP. It therefore trivially reduces to both problems in any norm in an approximation-preserving way (see Section 2.2 for definitions of these problems). In particular, our result not only implies new hardness for GapCRP$_p$ with $p < \infty$, but also for LinDisc in the ℓ_p norm for $p < \infty$ (LinDisc in finite ℓ_p norms does not seem to have been studied before).

Our main result is the following theorem. See also Figure 1.

▶ **Theorem 1** (NP-hardness of approximation for BinGapCRP$_p$)**.** *Let*

$$\gamma(p) := \left(\frac{9^p + 159 \cdot 3^p}{8^{p+2} + 96 \cdot 6^p}\right)^{1/p}, \tag{1}$$

and let $p_0 = 35.310188\ldots$ *be the unique value of* $p \geq 1$ *such that* $\gamma(p) = 1$*. Then for all* $p > p_0$ *and any constant* $\varepsilon > 0$*,* $(\gamma(p)-\varepsilon)$-BinGapCRP$_p$ *is* NP*-hard. Therefore,* $(\gamma(p)-\varepsilon)$-GapCRP$_p$ *and* $(\gamma(p)-\varepsilon)$-LinDisc$_p$ *for such* p *and* ε *are also* NP*-hard.*

We note that the expression $\gamma(p)$ in Equation (1) satisfies $\lim_{p\to\infty} \gamma(p) = 9/8$. Our second result shows how to achieve Π_2-hardness (rather than just NP-hardness) of $(9/8-\varepsilon)$-BinGapCRP in the ℓ_∞ norm.

▶ **Theorem 2** (Π_2-hardness of approximation for BinGapCRP$_\infty$)**.** *For any constant* $\varepsilon > 0$*,* $(9/8-\varepsilon)$-BinGapCRP$_\infty$ *is* Π_2*-hard.*

Theorem 2 is a strengthening of the main result in [29] – it shows hardness not just of $(9/8-\varepsilon)$-LinDisc$_\infty$ but of $(9/8-\varepsilon)$-BinGapCRP$_\infty$, where LinDisc$_\infty$ is the Linear Discrepancy Problem in the ℓ_∞ norm. Theorem 2 also gives an alternative proof of Π_2-hardness of approximation for γ-GapCRP$_\infty$, albeit with a smaller approximation factor than the one achieved by [20].

1.2 Related Work

Early work of McLoughlin [30] showed that the Covering Radius Problem on error-correcting codes is Π_2-hard by a reduction from a Π_2-hard variant of the 3-Dimensional Matching Problem, which is in turn shown to be Π_2-hard by a reduction from Π_2-3-SAT.[2] Twenty years later, work of Guruswami, Micciancio, and Regev [15] studied the complexity of the Covering Radius Problem both on codes and lattices. It showed a number of results, including that the Covering Radius Problem on codes is NP-hard to approximate to within any constant and Π_2-hard to approximate to within some constant. (It did not show hardness results for GapCRP on lattices.)

The work [15] also showed that 2-GapCRP$_p$ for any p is in AM via an elegant protocol. The fact that 2-GapCRP$_p$ is in AM provides evidence that γ-GapCRP$_p$ for $\gamma \geq 2$ is unlikely to be Π_2-hard, since if it were the polynomial hierarchy would collapse. However, being in AM says nothing about the potential NP-hardness of these problems. In particular it could very well be the case that γ-GapCRP is NP-hard for every constant $\gamma \geq 1$ as is the case for the Covering Radius Problem on codes.

[2] Here and throughout this section we refer to linear error-correcting codes simply as "codes."

The factor of $\gamma = 2$ in the [15] AM protocol follows generically for GapCRP in any norm from the triangle inequality, but it is not always tight. Follow-up work by Haviv, Lyubashevsky, and Regev [19] studied the distribution of the (normalized) ℓ_p distance of a random point in a fundamental domain of a lattice to the lattice. This distribution is closely related to the [15] AM protocol, and [19] showed that γ-GapCRP$_2$ for $\gamma > \sqrt{3} \approx 1.732$ is in AM under a certain conjecture about this distribution in the ℓ_2 norm. Their conjecture was subsequently proved by Magazinov [28].

Understanding the worst-case hardness of γ-GapCRP is also motivated by its relevance to cryptography. Indeed, [32] showed that γ-GapCRP with sufficiently large polynomial $\gamma = \gamma(n)$ can be taken as the starting worst-case problem for a worst-case to average-case reduction to the problems used as the foundation of lattice-based cryptography.

The Haviv-Regev hardness reduction

The work of Haviv and Regev [20] is the only prior work that studied the worst-case hardness of GapCRP on lattices. As noted above, they showed that for every constant $\varepsilon > 0$, $(3/2-\varepsilon)$-GapCRP$_\infty$ is Π_2-hard and that for all sufficiently large p there exists $\gamma = \gamma(p)$ such that γ-GapCRP$_p$ is Π_2-hard to approximate. However, the cutoff for "sufficiently large" and the approximation factors $\gamma(p)$ for $p < \infty$ are non-explicit.

The reduction in [20] is from the Group Coloring Problem on graphs with respect to the group $\mathbb{Z}_3$, a problem that is Π_2-hard.[3] The problem is defined as follows. The input is a directed graph $G = (V, E)$. The goal is to decide whether for every assignment $\phi : E \to \mathbb{Z}_3$ of labels to edges, there exists a coloring $c : V \to \mathbb{Z}_3$ of vertices such that for every edge $e = (u, v) \in E$, $c(u) - c(v) \neq \phi(e)$. The reduction to GapCRP is as follows. Let $A = A(G) \in \mathbb{Z}^{m \times n}$ for $m := |E|$, $n := |V|$ be the incidence matrix of G. That is, A is the matrix with two non-zero entries per row defined as

$$A_{i,j} := \begin{cases} -1 & \text{if edge } e_i = (v_j, u) \text{ for some } u \in V, \\ 1 & \text{if edge } e_i = (u, v_j) \text{ for some } u \in V, \\ 0 & \text{otherwise.} \end{cases}$$

The reduction then outputs (a basis of) the lattice $\mathcal{L} := A \cdot \mathbb{Z}^n + 3\mathbb{Z}^m$. One may also view A as the generator matrix of a ternary code $\mathcal{C} \subseteq \mathbb{F}_3^m$, and $\mathcal{L}$ as the Construction-A lattice $\mathcal{C} + 3\mathbb{Z}^m$.

To show NP-hardness of γ-GapCRP$_p$ in some ℓ_p norm (for large γ), one might try to reduce from variants of the 3-Coloring Problem, which is the NP analog of the Π_2-complete $\mathbb{Z}_3$-Group Coloring Problem. Indeed, strong hardness of approximation results are known for Almost-3-Coloring – the optimization variant of 3-Coloring in which the goal is to minimize the number of monochromatic edges – assuming variants of the Unique Games Conjecture [9]. However, it is unclear how to get this approach to work.

Combinatorial and linear discrepancy

The (combinatorial) *discrepancy* of a set system $S_1, \ldots, S_m \subseteq \{x_1, \ldots, x_n\}$ is defined as $\min_{x_1,\ldots,x_n \in \{-1,1\}} \max_{i \in [m]} |\sum_{j \in S_i} x_j|$, and the goal of the *combinatorial discrepancy problem* is, given the set system as input, to find $x_1, \ldots, x_n \in \{-1, 1\}$ that achieve this minimum.

[3] Their reduction actually works with $\mathbb{Z}_q$ for any $q \geq 3$, but they achieve the best hardness of approximation with $q = 3$.

Equivalently, if $\mathbf{A} \in \{0,1\}^{m \times n}$ is the incidence matrix of the input set system, the goal is to find $\boldsymbol{x} \in \{-1,1\}^n$ that minimizes $\|\mathbf{A}\boldsymbol{x}\|_\infty$. Combinatorial discrepancy and other variants of discrepancy are used widely in computer science and math (see, e.g., [7]).

Linear discrepancy was originally defined and related to other notions of discrepancy in work of Lovász, Spencer, and Vestergombi [27], and it appears as a key ingredient in certain approximation algorithms (see, e.g., [11, 37, 23]). Formally, we define the linear discrepancy $\operatorname{lindisc}(\mathbf{B})$ of a matrix $\mathbf{B}$ in the ℓ_p norm as

$$\operatorname{lindisc}_p(\mathbf{B}) := \max_{\boldsymbol{w} \in [0,1]^n} \min_{\boldsymbol{x} \in \{0,1\}^n} \|\mathbf{B}(\boldsymbol{w} - \boldsymbol{x})\|_p \ . \tag{2}$$

This definition is syntactically very similar to the definition of the covering radius in the ℓ_p norm, μ_p. The difference is that in the definition of μ_p, the minimum is taken over $\boldsymbol{x} \in \mathbb{Z}^n$ rather than $\boldsymbol{x} \in \{0,1\}^n$. See Sections 2.1 and 2.2.

Combinatorial discrepancy is captured, up to translation and scaling, by fixing $\boldsymbol{w} := \frac{1}{2} \cdot \mathbf{1}$ in Equation (2) instead of taking the max over all $\boldsymbol{w} \in [0,1]^n$ there. Linear discrepancy was first investigated from a complexity perspective by Li and Nikolov [26], who provided both algorithms and hardness results. In particular, they showed that the exact Linear Discrepancy Problem in the ℓ_∞ norm, LinDisc_∞, is NP-hard and that it is contained in Π_2. Their hardness result was subsequently improved by Manurangsi [29], who showed that γ-LinDisc_∞ is Π_2-hard for $\gamma := 9/8 - \varepsilon$ for any constant $\varepsilon > 0$. Our work makes extensive use of [29].

"Standard" linear discrepancy corresponds to $\operatorname{lindisc}_\infty$, and, to the best of our knowledge, our work is the first to extend the definition to finite ℓ_p norms. This extension is useful for our work. Moreover, we note that although linear discrepancy does not seem to have been studied in finite ℓ_p norms before, both combinatorial discrepancy and hereditary discrepancy have (see, e.g., [34]).

Lattice hardness results in large ℓ_p norms

We recall that our main hardness result in Theorem 1 only holds for GapCRP_p with $p > p_0 \approx 35.31$, whereas likely the most interesting case is $p = 2$. We note that it has been the case several times that progress on showing hardness of lattice problems in large ℓ_p norms has precipitated progress for the Euclidean case of $p = 2$. In particular, Khot [25] showed randomized NP-hardness of approximation for the Shortest Vector Problem (SVP) in large ℓ_p norms, and then shortly thereafter showed such hardness in all ℓ_p norms with $p > 1$ in [24]. More recently, Hair and Sahai [16] showed *deterministic* NP-hardness of approximation for SVP in ℓ_p norms for $p > 2$, and then showed hardness of approximation for SVP in the Euclidean norm [17].

More generally, quite a few works on the hardness of lattice problems have shown their strongest results in large ℓ_p norms (see, e.g., [1, 4, 5]).

1.3 Technical Overview

At a high level, our hardness reduction for BinGapCRP_p follows the hardness reduction for LinDisc_∞ in [29]. The reduction in [29] reduces from NAE-E3-SAT,[4] the constraint

[4] In fact, [29] reduces from the NAE-3-SAT problem (rather than the NAE-E3-SAT problem), in which constraints may have two literals (rather than always having three). The reduction simply doubles an arbitrary literal in any constraint with two literals, which leads to the "incidence" matrix $\mathbf{B}$ defined below potentially containing entries from $\{-2, 2\}$ as well as $\{-1, 0, 1\}$. In this work, we reduce from NAE-E3-SAT.

satisfaction problem in which each constraint is a function of three literals (Boolean variables v_j or their negations $\neg v_j$), and a constraint is satisfied by an assignment to its variables if and only if the assignment neither satisfies none of its literals nor all of its literals.

Specifically, on input a NAE-E3-SAT formula $\phi(v_1, \ldots, v_n)$ with n Boolean variables and with m constraints, the NP-hardness reduction in [29] first computes the constraint-variable incidence matrix $\mathbf{B} = \mathbf{B}(\phi) \in \mathbb{Z}^{m \times n}$ with entries

$$B_{i,j} := \begin{cases} 1 & \text{if constraint } i \text{ contains } v_j \ , \\ -1 & \text{if constraint } i \text{ contains } \neg v_j \ , \\ 0 & \text{otherwise .} \end{cases}$$

It then outputs a $\mathrm{LinDisc}_\infty$ instance with matrix

$$\mathbf{A} := \begin{pmatrix} \frac{1}{3}\mathbf{B} & \frac{1}{3}\mathbf{B} & \frac{1}{3}\mathbf{B} \\ & \mathbf{G} \otimes \mathbf{I}_n & \end{pmatrix} \in \mathbb{Q}^{(m+3n)\times 3n} \ , \tag{3}$$

where $\mathbf{G}$ is the 3×3 gadget matrix

$$\mathbf{G} := \begin{pmatrix} 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \end{pmatrix} \ ,$$

$\otimes$ is the Kronecker product and $\mathbf{I}_n$ is the $n \times n$ identity matrix.

Both the completeness and soundness analysis of the reduction in [29] crucially use properties of $\mathbf{G}$. For example, [29, Lemma 3] argues that $\mathbf{G}$ has low discrepancy by upper bounding the distance between $\mathbf{G}\boldsymbol{u}$ and $\mathbf{G} \cdot \{0,1\}^3$ for each $\boldsymbol{u} \in [0,1]^3$ via a case analysis on the ℓ_1 norm of $\boldsymbol{u}$. The soundness analysis shows that, if there exists a close vector in the set $\mathbf{A} \cdot \{0,1\}^{3n}$ to $\mathbf{A} \cdot (\frac{1}{2}\mathbf{1})$, then ϕ must be satisfiable. [29] also shows Π_2-hardness of $(9/8 - \varepsilon)$-$\mathrm{LinDisc}_\infty$ by outputting and analyzing a matrix $\mathbf{A}' = \mathbf{A}'(\phi)$ defined in terms of ϕ that is related to the matrix $\mathbf{A}$ defined in Equation (3) but more complicated. We use this same matrix $\mathbf{A}'$ (defined in Equation (27)) to prove Theorem 2.

At a technical level, our NP-hardness result adapts the machinery in [29] from ℓ_∞ to general ℓ_p norms, and shows that, if the NAE-E3-SAT formula ϕ is unsatisfiable, then not only is $\mathrm{lindisc}(\mathbf{A})$ large, in fact $\mu(\mathcal{L}(\mathbf{A}))$ is large. That is, we show that if ϕ is unsatisfiable then there exists $\boldsymbol{w} \in [0,1]^{3n}$ (in fact, $\boldsymbol{w} := \frac{1}{2} \cdot \mathbf{1}$) such that $\|\mathbf{A}(\boldsymbol{w} - \boldsymbol{x})\|_p$ is large not only for all $\boldsymbol{x} \in \{0,1\}^{3n}$ but for all $\boldsymbol{x} \in \mathbb{Z}^{3n}$. Doing these things requires making a number of modifications to the hardness reduction and analysis used by [29], such as:

1. Using *hardness of approximation* for NAE-E3-SAT with soundness $15/16 + \varepsilon$ and perfect completeness; see Theorem 10.
2. Taking the Kronecker product of G with the "ℓ_p norm degree" diagonal matrix

   $$\mathbf{D}_p := \mathbf{D}_p(\phi) = \mathrm{diag}(\deg(v_1)^{1/p}, \ldots, \deg(v_n)^{1/p}) \in \mathbb{R}^{n \times n}$$

   in the definition of $\mathbf{A}$ instead of just taking the Kronecker product with I_n. Here $\deg(v_i)$ counts the number of occurrences of the variable v_i and its negation $\neg v_i$ in the formula ϕ.
3. Extending the analysis from [29] for the gadget matrix $\mathbf{G}$. For example, we prove an ℓ_p analog of the fact that $\mathrm{lindisc}(\mathbf{G})_\infty$ is small (and in fact, that something stronger is true) in Lemma 11, and we observe that $\mathbf{G} \cdot \mathbf{0} = \mathbf{0}$ and $\mathbf{G} \cdot \mathbf{1} = \mathbf{1}$ are the only closest vectors in $\mathcal{L}(\mathbf{G}) = \mathbf{G} \cdot \mathbb{Z}^3$ to $\mathbf{G} \cdot (\frac{1}{2}\mathbf{1})$ (and not just the only closest vectors in $\mathbf{G} \cdot \{0,1\}^3$ to $\mathbf{G} \cdot (\frac{1}{2}\mathbf{1})$) in Lemma 12.

For Item 1, we note that the soundness-to-completeness ratio of 15/16 in Theorem 10 is *not* tight. In fact, recent work of Brakensiek, Huang, Potechin, and Zwick [6] shows tight upper and lower bounds for NAE-E3-SAT with a lower (i.e., better) soundness-to-completeness ratio of roughly $0.9089 < 15/16 = 0.9375$. However, Theorem 10 shows hardness of (δ, ε)-NAE-E3-SAT *with perfect completeness* (i.e., with $\varepsilon = 1$), whereas the result of [6] does not. This is important since our main hardness reduction for BinGapCRP_p requires reducing from (δ, ε)-NAE-E3-SAT with nearly perfect completeness. The hardness result in Theorem 10 appeared in work of Guruswami [14], which observed that it follows by combining a reduction due to Petrank [35] and a hardness result due to Håstad [18]. (We include a proof of this result in the full version.)

The reason for using $\mathbf{D}_p$ as in Item 2 is as follows. In our analysis, we analyze the quantity $\|\mathbf{A}\boldsymbol{y}\|_p^p$ for coefficient vectors $\boldsymbol{y} \in \mathbb{Z}^{3n}$ expressed as a sum indexed by the m constraints in ϕ (see Lemma 13). We wish to penalize vectors $\boldsymbol{y}$ that do not correspond to valid assignments to ϕ by having $\|\mathbf{A}\boldsymbol{y}\|_p^p$ be larger for such $\boldsymbol{y}$. Using the matrix $\mathbf{D}_p$ allows us to do this. Specifically, it allows us to balance the contributions of the top part $\begin{pmatrix}\frac{1}{3}\mathbf{B} & \frac{1}{3}\mathbf{B} & \frac{1}{3}\mathbf{B}\end{pmatrix} \cdot \boldsymbol{y}$ and the bottom part $(\mathbf{G} \otimes \mathbf{D}_p) \cdot \boldsymbol{y}$ of $\mathbf{A}\boldsymbol{y}$ to each term in this sum. The top part's contribution to a summand will be large if $\boldsymbol{y}$ corresponds to a valid but unsatisfying assignment to ϕ, and the bottom part's contribution to a summand will be large if $\boldsymbol{y}$ corresponds to an invalid assignment to ϕ.

1.4 Open Questions

We again emphasize the central open question that we consider in this work: For which values of $\gamma = \gamma(p)$ is γ-GapCRP in the ℓ_p norm NP-hard, and for which is it Π_2-hard? In particular, we reiterate that the problem of showing NP-hardness of exact GapCRP in the ℓ_2 norm (the most important special case) remains open. We again note that the AM protocol of [15] for approximate GapCRP (conditionally) rules out Π_2-hardness for γ-GapCRP_p for $\gamma \geq 2$, but says nothing about NP-hardness of γ-GapCRP_p. Manurangsi [29] notes that no such analogous AM protocol is known for LinDisc, and asks if one exists.

2 Preliminaries

We will use boldface, lowercase letters to denote vectors. We will sometimes abuse notation when defining column vectors in terms of other column vectors by writing things like $\boldsymbol{x} := (\boldsymbol{y}, \boldsymbol{z})$ to mean $\boldsymbol{x} := (\boldsymbol{y}^T, \boldsymbol{z}^T)^T$. We will use $\mathbf{0}$ and $\mathbf{1}$ to denote the all-0s and all-1s vectors, respectively.

We define the predicate $\text{sgndiff}(\boldsymbol{x}, \boldsymbol{y})$ on vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ to be true if and only if the Hadamard (entry-wise) product $\boldsymbol{x} \odot \boldsymbol{y} \in \mathbb{R}^n$ has at least one non-positive coordinate and at least one non-negative coordinate. We will use the simple fact that if $\text{sgndiff}(\boldsymbol{x}, \boldsymbol{y})$ holds and if $\max_i |x_i y_i| \leq r$ for some $r \geq 0$, then $|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| = |\sum_{i=1}^n x_i y_i| \leq r(n-1)$.

2.1 Covering Radius and Linear Discrepancy

We call a matrix $\mathbf{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n) \in \mathbb{R}^{m \times n}$ with full column rank a *basis*. In particular, $m \geq n$ for a basis. The lattice $\mathcal{L}(\mathbf{B})$ generated by $\mathbf{B}$ is defined as

$$\mathcal{L}(\mathbf{B}) := \left\{ \sum_{i=1}^n a_i \boldsymbol{b}_i : a_1, \ldots, a_n \in \mathbb{Z} \right\} .$$

For $p \geq 1$, the ℓ_p *covering radius* of a lattice $\mathcal{L} \subset \mathbb{R}^m$ is defined as

$$\mu_p(\mathcal{L}) := \max_{\boldsymbol{t} \in \mathrm{span}(\mathcal{L})} \mathrm{dist}_p(\boldsymbol{t}, \mathcal{L}) \ ,$$

where for a discrete set S, we define $\mathrm{dist}_p(\boldsymbol{t}, S) := \min_{\boldsymbol{y} \in S} \|\boldsymbol{t} - \boldsymbol{y}\|_p$. The *span* $\mathrm{span}(\mathcal{L})$ of a lattice $\mathcal{L}$ is the linear span of the vectors in the lattice (or equivalently, the linear span $\mathrm{span}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ of the vectors in a basis $\mathbf{B} := (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ of $\mathcal{L}$). Because $\mathrm{dist}_p(\boldsymbol{t} + \boldsymbol{y}, \mathcal{L}) = \mathrm{dist}_p(\boldsymbol{t}, \mathcal{L})$ for any $\boldsymbol{t} \in \mathbb{R}^m$ and $\boldsymbol{y} \in \mathcal{L}$, we have that if $\mathcal{L} = \mathcal{L}(\mathbf{B})$ for a basis $\mathbf{B} \in \mathbb{R}^{m \times n}$,

$$\mu_p(\mathcal{L}) = \max_{\boldsymbol{w} \in \mathbb{R}^n} \min_{\boldsymbol{x} \in \mathbb{Z}^n} \|\mathbf{B}(\boldsymbol{w} - \boldsymbol{x})\|_p = \max_{\boldsymbol{w} \in [0,1]^n} \min_{\boldsymbol{x} \in \mathbb{Z}^n} \|\mathbf{B}(\boldsymbol{w} - \boldsymbol{x})\|_p \ . \tag{4}$$

That is, because the distance of a vector $\mathbf{B}\boldsymbol{w}$ to $\mathcal{L}(\mathbf{B})$ is invariant under shifts by lattice vectors (i.e., vectors $\mathbf{B}\boldsymbol{x}$ for $\boldsymbol{x} \in \mathbb{Z}^n$), it suffices to consider the maximum distance of vectors $\mathbf{B}\boldsymbol{w}$ in the fundamental parallelepiped $\mathcal{P}(\mathbf{B}) := \mathbf{B} \cdot [0,1]^n$ to the lattice.

We will also work with *linear discrepancy*, a quantity closely related to the covering radius that was first defined in [27]. For $p \geq 1$, the linear discrepancy $\mathrm{lindisc}(\mathbf{B})$ of a matrix $\mathbf{B}$ is defined as[5]

$$\mathrm{lindisc}_p(\mathbf{B}) := \max_{\boldsymbol{w} \in [0,1]^n} \min_{\boldsymbol{x} \in \{0,1\}^n} \|\mathbf{B}(\boldsymbol{w} - \boldsymbol{x})\|_p \ . \tag{5}$$

Covering radius versus linear discrepancy

We note that the equivalent formulation of $\mu_p(\mathcal{L})$ on the right-hand side of Equation (4) and the definition of $\mathrm{lindisc}(\mathbf{B})$ are syntactically very similar. The only difference is that the minimum in Equation (4) is taken over $\boldsymbol{x} \in \mathbb{Z}^n$ (which corresponds to the entire lattice $\mathcal{L}(\mathbf{B}) = \{\mathbf{B}\boldsymbol{x} : \boldsymbol{x} \in \mathbb{Z}^n\}$) whereas in Equation (5) the minimum is only taken over $\boldsymbol{x} \in \{0,1\}^n$ (which corresponds to *binary* linear combinations $\mathbf{B}\boldsymbol{x}$ of basis vectors).

Despite the similarity of the definitions of μ_p and $\mathrm{lindisc}_p$, we emphasize that they capture different quantities. In particular, the closest vector in the lattice $\mathcal{L}(\mathbf{B}) := \mathbf{B} \cdot \mathbb{Z}^n$ to a vector $\mathbf{B} \cdot \boldsymbol{w}$ for $\boldsymbol{w} \in [0,1]^n$ need not be a vector in $\mathbf{B} \cdot \{0,1\}^n$. We give a simple example of this for the lattice $\mathbb{Z}^2$ when the input basis $\mathbf{B}$ consists of rather long vectors.

▶ **Example 3.** Consider the basis

$$\mathbf{B} := \begin{pmatrix} 3 & 4 \\ 1 & 1 \end{pmatrix} \ ,$$

and note that $\mathcal{L}(\mathbf{B}) = \mathbb{Z}^2$ since $\det(\mathbf{B}) = -1$ (i.e., $\mathbf{B}$ is unimodular). Let $\boldsymbol{w} := (1, 1/2)^T$, and note that $\mathbf{B}\boldsymbol{w} = (5, 3/2)^T$. Then

$$\mathrm{dist}_p(\mathbf{B}\boldsymbol{w}, \mathcal{L}(\mathbf{B})) = \mathrm{dist}_p(\mathbf{B}\boldsymbol{w}, \mathbb{Z}^2) = \|\mathbf{B}((3, -1)^T - \boldsymbol{w})\|_p = 1/2 \ ,$$

but one can check that $\|\mathbf{B}(\boldsymbol{x} - \boldsymbol{w})\|_p > 1/2$ for all $\boldsymbol{x} \in \{0,1\}^2$.

2.2 Covering Radius and Linear Discrepancy as Computational Problems

We next define the approximate decisional Covering Radius Problem in the ℓ_p norm, γ-GapCRP, and the closely related approximate Linear Discrepancy Problem in the ℓ_p norm, γ-$\mathrm{LinDisc}_p$.

[5] The definition of $\mathrm{lindisc}(\mathbf{B})$ does not require $\mathbf{B}$ to be a basis. I.e., $\mathbf{B}$ is allowed to have linearly dependent columns.

▶ **Definition 4.** *Let $p \in [1, \infty]$ and let $\gamma = \gamma(n) \geq 1$. The* γ-approximate decisional Covering Radius Problem in the ℓ_p norm *(γ-GapCRP$_p$), is the problem where, on input a basis $\mathbf{B} \in \mathbb{Q}^{m \times n}$ and a value $r > 0$, the goal is to distinguish between the following two cases.*

- *(YES instance.)* $\mu_p(\mathcal{L}(\mathbf{B})) \leq r$.
- *(NO instance.)* $\mu_p(\mathcal{L}(\mathbf{B})) > \gamma r$.

▶ **Definition 5.** *Let $p \in [1, \infty]$ and let $\gamma = \gamma(n) \geq 1$. The* γ-approximate decisional Linear Disrepancy Problem in the ℓ_p norm *(γ-LinDisc$_p$) is the problem where, on input a basis $\mathbf{B} \in \mathbb{Q}^{m \times n}$ and a value $r > 0$, the goal is to distinguish between the following two cases.*

- *(YES instance.)* $\text{lindisc}_p(\mathbf{B}) \leq r$.
- *(NO instance.)* $\text{lindisc}_p(\mathbf{B}) > \gamma r$.

Finally, we define the binary decisional Covering Radius Problem BinGapCRP, which has the YES instances of LinDisc and the NO instances of GapCRP.

▶ **Definition 6.** *Let $p \in [1, \infty]$ and let $\gamma = \gamma(n) \geq 1$. The* γ-approximate decisional Binary Covering Radius Problem in the ℓ_p norm *(γ-BinGapCRP$_p$) is the problem where, on input a basis $\mathbf{B} \in \mathbb{Q}^{m \times n}$ and a value $r > 0$, the goal is to distinguish between the following two cases.*

- *(YES instance.)* $\text{lindisc}_p(\mathbf{B}) \leq r$.
- *(NO instance.)* $\mu_p(\mathcal{L}(\mathbf{B})) > \gamma r$.

We note that $\mu_p(\mathcal{L}(\mathbf{B})) \leq \text{lindisc}_p(\mathbf{B})$ for any $p \in [1, \infty]$ and any basis $\mathbf{B}$. Therefore, for any $p \in [1, \infty]$ and $\gamma \geq 1$, YES (respectively, NO) instances of γ-BinGapCRP$_p$ are YES (respectively, NO) instances of both γ-GapCRP$_p$ and γ-LinDisc$_p$. We therefore immediately have the following observation.

▶ **Proposition 7.** *For any $\gamma \geq 1$ and $p \in [1, \infty]$, the identity mapping $(\mathbf{B}, r) \mapsto (\mathbf{B}, r)$ is a (polynomial-time) reduction from* γ-BinGapCRP$_p$ *to* γ-GapCRP$_p$ *and to* γ-LinDisc$_p$.

2.3 Constraint Satisfaction Problems

A (Boolean) *constraint satisfaction problem* (CSP) is defined by a set of Boolean variables $X = \{x_1, \ldots, x_n\}$, and a set of functions called *constraints* $\mathcal{C} = \{C_1, \ldots, C_m\}$ with $C_i \colon \{0,1\}^{|C_i|} \to \{0,1\}$. An *assignment* of the variables is a map from $X \to \{0,1\}$. A constraint C_i is *satisfied* by an assignment if the output of the constraint on the assignment of the variables is 1. Let ϕ be a CSP and let ψ be an assignment of the variables in ϕ. We define $\text{val}_\psi(\phi)$ to be the fraction of constraints in ϕ satisfied by an assignment ψ, and define the *value* of ϕ to be $\text{val}(\phi) := \max_\psi \text{val}_\psi(\phi)$. We then define the computational problem (δ, ε)-CSP as follows.

▶ **Definition 8** (Constraint Satisfaction Problems (CSP))**.** *For $\delta, \varepsilon \in \mathbb{R}$ be such that $0 < \delta \leq \varepsilon \leq 1$, (δ, ε)-CSP is the decision problem defined as follows. On input a collection of constraints $\mathcal{C} = \{C_1, \ldots, C_m\}$, the goal is to decide between the follows two cases.*

- *(YES instance.)* $\text{val}(\phi) \geq \varepsilon$.
- *(NO instance.)* $\text{val}(\phi) < \delta$.

We now introduce NAE-Ek-SAT, a k-ary CSP that will be important for our hardness reductions.

▶ **Definition 9** ((δ, ε)-NAE-Ek-SAT problem)**.** *A* NAE-Ek-SAT *formula ϕ consists of n Boolean variables $X = \{x_1, \ldots, x_n\}$ and m constraints $C_1, \ldots, C_m$, where each constraint C_i is a function of exactly k distinct literals (Boolean variables x_j or their negations $\neg x_j$). A constraint C_i is satisfied by an assignment if and only if not all k of its literals evaluate either to 1 or to 0.*

Let $\delta, \varepsilon \in \mathbb{R}$ be such that $0 < \delta \leq \varepsilon \leq 1$. The goal of the (δ, ε)-NAE-Ek-SAT problem is, given a NAE-Ek-SAT *formula as input, to decide between the following two cases.*

- *(YES instance.)* $\mathrm{val}(\phi) \geq \varepsilon$.
- *(NO instance.)* $\mathrm{val}(\phi) < \delta$.

We represent NAE-Ek-SAT constraints C as sets of k literals, and we use the notation $\mathrm{vars}(C)$ to denote the set of indices of variables involved in a NAE-E3-SAT constraint C. For example, if $k = 3$ and $C := \{v_1, \neg v_3, \neg v_4\}$ then $\mathrm{vars}(C) = \{1, 3, 4\}$. We define $\mathrm{sgn}(\ell) \in \{-1, 1\}$ and $\mathrm{var}(\ell)$ to be the sign and the index of the variable underlying a literal ℓ, respectively. For example, $\mathrm{sgn}(\neg v_4) = -1$ and $\mathrm{var}(\neg v_4) = 4$.

We next give a result showing NP-hardness of $(15/16+\varepsilon, 1)$-NAE-E3-SAT for any constant $\varepsilon > 0$ in Theorem 10. This hardness result appeared in work of Guruswami [14], which observed that it follows by combining a reduction due to Petrank [35] and a hardness result due to Håstad [18].

▶ **Theorem 10.** *For any constant $\varepsilon \in (0, 1/16)$, $(15/16 + \varepsilon, 1)$-NAE-E3-SAT is NP-hard.*

See our full version for a proof of Theorem 10.

3 Reduction from NAE-E3-SAT to BinaryGapCRP

3.1 The Reduction

We start by giving our reduction from (δ, ε)-NAE-E3-SAT to γ-BinGapCRP$_p$ for certain values of $\delta, \varepsilon, \gamma, p$. In what follows, we will analyze the reduction and for which parameter values it is valid. We emphasize again that the reduction is similar to the NP-hardness reduction in [29] for $\mathrm{LinDisc}_\infty$.

Let ϕ be the input (δ, ε)-NAE-E3-SAT formula, and assume that ϕ has n variables $v_1, \ldots, v_n$ and m constraints $C_1, \ldots, C_m$. We define the incidence matrix $\mathbf{B} \in \mathbb{R}^{m \times n}$ of ϕ as

$$B_{i,j} := B_{i,j}(\phi) = \begin{cases} 1 & \text{if constraint } C_i \text{ contains literal } v_j \ , \\ -1 & \text{if constraint } C_i \text{ contains literal } \neg v_j \ , \\ 0 & \text{otherwise .} \end{cases} \tag{6}$$

We will also use the following gadget matrix, defined in [29], which we analyze in Section 3.2.

$$\mathbf{G} := \begin{pmatrix} 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \end{pmatrix} . \tag{7}$$

Let $\deg_\phi(v_j)$ denote the number of constraints C_i in ϕ containing either v_j or $\neg v_j$. When the formula ϕ is clear from context, we simply write $\deg(v_j)$. For $p \in [1, \infty]$, define the $n \times n$ diagonal matrix

$$\mathbf{D}_p := \mathbf{D}_p(\phi) = \mathrm{diag}(\deg(v_1)^{1/p}, \ldots, \deg(v_n)^{1/p}) = \begin{pmatrix} \deg(v_1)^{1/p} & & \\ & \ddots & \\ & & \deg(v_n)^{1/p} \end{pmatrix} .$$

Finally, we output the γ-BinGapCRP$_p$ instance consisting of the matrix

$$\mathbf{A} := \mathbf{A}_p(\phi) = \begin{pmatrix} \frac{1}{3}\mathbf{B} & \frac{1}{3}\mathbf{B} & \frac{1}{3}\mathbf{B} \\ & \mathbf{G} \otimes \mathbf{D}_p & \end{pmatrix} \in \mathbb{R}^{(m+3n) \times 3n} \ , \tag{8}$$

and distance threshold

$$r := (\varepsilon m \cdot (4/3)^p + (1 - \varepsilon)m \cdot 2^p + 3m(2 + (4/3)^p))^{1/p} > 0 \ . \tag{9}$$

3.2 Analysis of the matrix G

We begin by presenting some technical lemmas about $\mathbf{G}$ that will be useful in our hardness reductions. We start with Lemma 11, which adapts [29, Lemma 3] from the ℓ_∞ norm to ℓ_p norms.

▶ **Lemma 11.** *For any $\boldsymbol{u} \in [0,1]^3$ and $b \in \{-1,1\}$, there exists $\boldsymbol{z} \in \{0,1\}^3$ such that*

1. *(Low discrepancy w.r.t. $\mathbf{G}$.) $\|\mathbf{G}(\boldsymbol{u}-\boldsymbol{z})\|_p^p \leq 2 + (4/3)^p$, where $\mathbf{G}$ is as defined in Equation (7).*
2. *(Sign agreement.) $b \cdot (\mathbf{1}^T(\boldsymbol{u}-\boldsymbol{z})) \geq 0$.*
3. *(Low discrepancy w.r.t. $\mathbf{1}^T$.) $|\mathbf{1}^T(\boldsymbol{u}-\boldsymbol{z})| \leq 2$.*

Proof. The second and third properties are identical to the corresponding properties in [29, Lemma 3] (which are proved there), and so we only prove the first property. We follow the structure of the proof of [29, Lemma 3, Item 1] to prove the first property.

We assume without loss of generality that $0 \leq u_1 \leq u_2 \leq u_3 \leq 1$ throughout the proof. This assumption is justified by the easy-to-check fact that for all 3×3 permutation matrices $\mathbf{P}$ and $\boldsymbol{x} \in \mathbb{R}^3$, $\|\mathbf{G}\boldsymbol{x}\|_\infty = \|\mathbf{G}\mathbf{P}\boldsymbol{x}\|_\infty$.

We now proceed to the proof of Item 1. We will consider three cases, and in each case we will bound each coordinate of $\mathbf{G}(\boldsymbol{u}-\boldsymbol{z})$ individually.

Case 1: $\mathbf{1}^T\boldsymbol{u} \geq 2$. In this case we set $\boldsymbol{z} := (0,1,1)^T$, and have

$$\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}) = \begin{pmatrix} u_1+u_2-u_3 \\ u_1-u_2+u_3 \\ -u_1+u_2+u_3-2 \end{pmatrix}.$$

We have following bounds:

- $-1 \leq -u_3 \leq (\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}))_1 = (\mathbf{G}\boldsymbol{u})_1 \leq u_1 \leq 1$.
- $-1 \leq -u_2 \leq (\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}))_2 = (\mathbf{G}\boldsymbol{u}))_2 \leq u_3 \leq 1$.
- $-4/3 \leq u_3 - 2 \leq (\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}))_3 = (\mathbf{G}\boldsymbol{u})_3 - 2 \leq -u_1 \leq 0$.

Case 2: $\mathbf{1}^T\boldsymbol{u} < 2$ and $(\mathbf{G}\boldsymbol{u})_3 \leq 4/3$.[6] In this case we set $\boldsymbol{z} := \mathbf{0}$, and have

$$\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}) = \begin{pmatrix} u_1+u_2-u_3 \\ u_1-u_2+u_3 \\ -u_1+u_2+u_3 \end{pmatrix}.$$

We have the following bounds:

- $-1 \leq -u_3 \leq (\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}))_1 = (\mathbf{G}\boldsymbol{u})_1 \leq u_1 \leq 1$.
- $-1 \leq -u_2 \leq (\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}))_2 = (\mathbf{G}\boldsymbol{u})_2 \leq u_3 \leq 1$.
- $0 \leq u_3 \leq (\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}))_3 = (\mathbf{G}\boldsymbol{u})_3 \leq 4/3$.

Case 3: $\mathbf{1}^T\boldsymbol{u} < 2$ and $(\mathbf{G}\boldsymbol{u})_3 > 4/3$. In this case we set $\boldsymbol{z} := (0,0,1)^T$, and have

$$\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}) = \begin{pmatrix} u_1+u_2-u_3+1 \\ u_1-u_2+u_3-1 \\ -u_1+u_2+u_3-1 \end{pmatrix}.$$

We have the following bounds:

[6] In fact, the assumption that $\mathbf{1}^T\boldsymbol{u} < 2$ is not even necessary here.

- $0 \le -u_3 + 1 \le (\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}))_1 = (\mathbf{G}\boldsymbol{u})_1 + 1 \le u_1 + 1 = \frac{1}{2}(\mathbf{1}^T\boldsymbol{u} - (\mathbf{G}\boldsymbol{u})_3) + 1 < 1/3 + 1 = 4/3.$
- $-1 \le u_1 - 1 \le (\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}))_2 = (\mathbf{G}\boldsymbol{u})_2 - 1 \le u_3 - 1 \le 0.$
- $-1 \le u_3 - 1 \le (\mathbf{G}(\boldsymbol{u}-\boldsymbol{z}))_3 = (\mathbf{G}\boldsymbol{u})_3 - 1 \le \mathbf{1}^T\boldsymbol{u} - 1 < 1.$

In each case, we have $\|\mathbf{G}(\boldsymbol{u}-\boldsymbol{z})\|_p^p \le 2 + (4/3)^p$, as needed. ◀

Lemma 12 is the other part of the extension of [29, Lemma 3], and facilitates a reduction to BinGapCRP, rather than just LinDisc. We both adapt [29, Lemma 3] to ℓ_p norms for $p < \infty$ and show that $\mathbf{G}(1/2 \cdot \mathbf{1}) = (1/2) \cdot \mathbf{1}$ is far from all vectors in $\mathcal{L}(\mathbf{G}) \setminus \{\mathbf{0}, \mathbf{1}\} = \mathbf{G} \cdot (\mathbb{Z}^3 \setminus \{\mathbf{0}, \mathbf{1}\})$ rather than just vectors in $\mathbf{G} \cdot (\{0,1\}^3 \setminus \{\mathbf{0}, \mathbf{1}\})$.

▶ **Lemma 12.** *Let $\boldsymbol{z} \in \mathbb{Z}^3$ and $p \in [1, \infty]$. Then we have the following:*

1. *For $\boldsymbol{z} \in \{\mathbf{0}, \mathbf{1}\}$, $\mathbf{G}(1/2 \cdot \mathbf{1} - \boldsymbol{z}) \in \{-1/2, 1/2\}^3$. Therefore, $\|\mathbf{G}(1/2 \cdot \mathbf{1} - \boldsymbol{z})\|_p^p = 3/2^p$ if $p < \infty$ and $\|\mathbf{G}(1/2 \cdot \mathbf{1} - \boldsymbol{z})\|_\infty = 1/2$.*
2. *For $\boldsymbol{z} \in \mathbb{Z}^3 \setminus \{\mathbf{0}, \mathbf{1}\}$, $\|\mathbf{G}(1/2 \cdot \mathbf{1} - \boldsymbol{z})\|_p^p \ge (2 + 3^p)/2^p$ if $p < \infty$ and $\|\mathbf{G}(1/2 \cdot \mathbf{1} - \boldsymbol{z})\|_\infty \ge 3/2$.*

Proof. We note that $\mathbf{G}((1/2) \cdot \mathbf{1} - \boldsymbol{z}) = (1/2) \cdot \mathbf{1} - \mathbf{G}\boldsymbol{z}$ so that[7]

$$\{(\mathbf{G}(1/2 \cdot \mathbf{1} - \boldsymbol{z}) : \boldsymbol{z} \in \mathbb{Z}^3\} = \{\mathbf{G}\boldsymbol{z} : \boldsymbol{z} \in \mathbb{Z}^3\} + (1/2) \cdot \mathbf{1} \subseteq (\mathbb{Z} + 1/2)^3 \ .$$

That is, for any $\boldsymbol{z} \in \mathbb{Z}^3$, each coordinate of $G((1/2 \cdot \mathbf{1} - \boldsymbol{z})$ is contained in $\mathbb{Z} + 1/2$. Fix such a coefficient vector $\boldsymbol{z} \in \mathbb{Z}^3$. Then for $p < \infty$, $\|\mathbf{G}((1/2) \cdot \mathbf{1} - \boldsymbol{z})\|_p^p = 3/2^p$ if $\mathbf{G}\boldsymbol{z} \in \{0,1\}^3$ and $\|\mathbf{G}((1/2) \cdot \mathbf{1} - \boldsymbol{z})\|_p^p \ge (2 + 3^p)/2^p$ otherwise. Similarly, $\|\mathbf{G}((1/2) \cdot \mathbf{1} - \boldsymbol{z})\|_\infty = 1/2$ if $\mathbf{G}\boldsymbol{z} \in \{0,1\}^3$ and $\|\mathbf{G}((1/2) \cdot \mathbf{1} - \boldsymbol{z})\|_\infty \ge 3/2$ otherwise. The lemma then follows by noting that $\mathcal{L}(\mathbf{G}) \cap \{0,1\}^3 = \{\mathbf{0}, \mathbf{1}\}$. ◀

3.3 Completeness Analysis

We start with an expression for the value of $\|\mathbf{A}(\boldsymbol{w} - \boldsymbol{x})\|_p^p$ for the matrix $\mathbf{A} = \mathbf{A}(\phi)$ defined in Equation (8) in terms of a NAE-E3-SAT formula ϕ and vectors $\boldsymbol{w}, \boldsymbol{x}$.

▶ **Lemma 13.** *Let $p \ge 1$, and for $i \in [3]$, let $\boldsymbol{y}^i \in \mathbb{R}^n$, let ϕ be an* NAE-E3-SAT *formula with constraints $C_1, \ldots, C_m$, and let $\mathbf{A} := \mathbf{A}_p(\phi)$ as defined in Equation (8). Let $\boldsymbol{y} := (\boldsymbol{y}^1, \boldsymbol{y}^2, \boldsymbol{y}^3) \in \mathbb{R}^{3n}$, let $\boldsymbol{y}^{sum} := \boldsymbol{y}^1 + \boldsymbol{y}^2 + \boldsymbol{y}^3 \in \mathbb{R}^n$, and for $j \in [n]$, let $\boldsymbol{y}_j := (y_j^1, y_j^2, y_j^3) \in \mathbb{R}^3$. Then*

$$\|\mathbf{A}\boldsymbol{y}\|_p^p = (1/3^p) \cdot \|\mathbf{B}\boldsymbol{y}^{sum}\|_p^p + \sum_{j=1}^{n} \deg(v_j) \cdot \left\|\mathbf{G}\boldsymbol{y}_j\right\|_p^p \ . \tag{10}$$

Let $\boldsymbol{x}^1, \boldsymbol{x}^2, \boldsymbol{x}^3 \in \mathbb{R}^n$, let $\boldsymbol{x} := (\boldsymbol{x}^1, \boldsymbol{x}^2, \boldsymbol{x}^3) \in \mathbb{R}^{3n}$, and let $\boldsymbol{x}^{sum} := \boldsymbol{x}^1 + \boldsymbol{x}^2 + \boldsymbol{x}^3 \in \mathbb{R}^n$. Furthermore, for $j \in [n]$, let $\boldsymbol{x}_j := (x_j^1, x_j^2, x_j^3)^T \in \mathbb{R}^3$, and let $\boldsymbol{b}_i$ denote the ith row of $\mathbf{B}$. Then in particular,

$$\left\|\mathbf{A}\left(\frac{1}{2} \cdot \mathbf{1} - \boldsymbol{x}\right)\right\|_p^p = \sum_{i=1}^{m} \left(\left|\frac{1}{3}\boldsymbol{b}_i\left(\frac{3}{2} \cdot \mathbf{1} - \boldsymbol{x}^{sum}\right)\right|^p + \sum_{j \in \mathrm{vars}(C_i)} \left\|\mathbf{G}\left(\frac{1}{2} \cdot \mathbf{1} - \boldsymbol{x}_j\right)\right\|_p^p \right) \ . \tag{11}$$

[7] This is equivalent to the fact that $\mathbf{G}$ has $\mathbf{1}$ as an eigenvector with corresponding eigenvalue 1.

Proof. For Equation (10),

$$\begin{aligned}\|\mathbf{A}(\boldsymbol{y})\|_p^p &= \left\|(1/3)\cdot(\mathbf{B}\boldsymbol{y}^1+\mathbf{B}\boldsymbol{y}^2+\mathbf{B}\boldsymbol{y}^3)\right\|_p^p + \|(\mathbf{G}\otimes\mathbf{D}_p)\cdot\boldsymbol{y}\|_p^p \\ &= \|(1/3)\cdot\mathbf{B}\boldsymbol{y}^{\text{sum}}\|_p^p + \sum_{j=1}^n \deg(v_j)\cdot\left\|\mathbf{G}\boldsymbol{y}_j\right\|_p^p .\end{aligned}$$

For Equation (11), we set $\boldsymbol{y} := \frac{1}{2}\cdot\mathbf{1}-\boldsymbol{x}$ and continue by re-grouping some terms.

$$\begin{aligned}\left\|\mathbf{A}\left(\frac{1}{2}\cdot\mathbf{1}-\boldsymbol{x}\right)\right\|_p^p &= \frac{1}{3^p}\cdot\left\|\mathbf{B}\left(\frac{3}{2}\cdot\mathbf{1}-\boldsymbol{x}^{\text{sum}}\right)\right\|_p^p + \sum_{j=1}^n \deg(v_j)\cdot\left\|\mathbf{G}\left(\frac{1}{2}\cdot\mathbf{1}-\boldsymbol{x}_j\right)\right\|_p^p \\ &= \frac{1}{3^p}\cdot\sum_{i=1}^m\left|\boldsymbol{b}_i\left(\frac{3}{2}\cdot\mathbf{1}-\boldsymbol{x}^{\text{sum}}\right)\right|^p + \sum_{j=1}^n \deg(v_j)\cdot\left\|\mathbf{G}\left(\frac{1}{2}\cdot\mathbf{1}-\boldsymbol{x}_j\right)\right\|_p^p \\ &= \sum_{i=1}^m\left(\left|\frac{1}{3}\boldsymbol{b}_i\left(\frac{3}{2}\cdot\mathbf{1}-\boldsymbol{x}^{\text{sum}}\right)\right|^p + \sum_{j\in\text{vars}(C_i)}\left\|\mathbf{G}\left(\frac{1}{2}\cdot\mathbf{1}-\boldsymbol{x}_j\right)\right\|_p^p\right).\end{aligned}$$

The first equality follows by setting $\boldsymbol{w} := 1/2\cdot\mathbf{1}$ in Equation (10). The third equality follows by noting for any expression $K(j)$ depending on j,

$$\sum_{j=1}^n \deg(v_j)K(j) = \sum_{j=1}^n \sum_{\text{vars}(C_i):x_j\in\text{vars}(C_i)} K(j) = \sum_{i=1}^m \sum_{j:x_j\in\text{vars}(C_i)} K(j) . \tag{12}$$

Indeed, one can view the two double sums as counting constraint-variable incidences in two different ways. ◀

We now state lemmas that will be used in the proof of Theorem 17. We begin with a lemma for completeness in which we upper bound $\text{lindisc}_p(\mathbf{A})$.

▶ **Lemma 14.** *Let $p\in[1,\infty)$, let $\varepsilon\in[0,1]$, let ϕ be an* NAE-E3-SAT *instance with m constraints and $\text{val}(\phi)\geq\varepsilon$, and let $\mathbf{A}:=\mathbf{A}_p(\phi)$ be defined as in Equation* (8)*. Then*

$$\text{lindisc}_p(\mathbf{A})^p \leq (\varepsilon\cdot(4/3)^p + (1-\varepsilon)\cdot 2^p + 3(2+(4/3)^p))\cdot m . \tag{13}$$

Proof. Let ψ be an assignment to the variables of ϕ such that $\text{val}_\psi(\phi)\geq\varepsilon$. Let $\boldsymbol{w}^1,\boldsymbol{w}^2,\boldsymbol{w}^3\in[0,1]^n$, and let $\boldsymbol{w} := (\boldsymbol{w}^1,\boldsymbol{w}^2,\boldsymbol{w}^3)$. We define $\boldsymbol{x} := (\boldsymbol{x}^1,\boldsymbol{x}^2,\boldsymbol{x}^3)$ for $\boldsymbol{x}^1,\boldsymbol{x}^2,\boldsymbol{x}^3\in\{0,1\}^n$ as follows. For each $j\in[n]$, the vector $\boldsymbol{x}_j := (x_j^1,x_j^2,x_j^3)^T$ is the vector $\boldsymbol{z}$ guaranteed by Lemma 11 applied to the vector $\boldsymbol{u} := \boldsymbol{w}_j = (w_j^1,w_j^2,w_j^3)^T$ and sign $b := 1-2\psi(v_j)$.

We begin by upper bounding $\|\mathbf{A}(\boldsymbol{w}-\boldsymbol{x})\|_p^p$, which we do by upper bounding the terms on the right-hand side of Equation (10) individually with $\boldsymbol{y} := \boldsymbol{w}-\boldsymbol{x}\in\mathbb{R}^{3n}$. From Lemma 11, Item 1 and the fact that ϕ has total degree $\sum_{j=1}^n\deg(v_j)=3m$, we have

$$\sum_{j=1}^n \deg(v_j)\cdot\|\mathbf{G}(\boldsymbol{w}_j-\boldsymbol{x}_j)\|_p^p \leq \sum_{j=1}^n \deg(v_j)\cdot(2+(4/3)^p) = 3m(2+(4/3)^p). \tag{14}$$

To upper bound $\left\|\frac{1}{3}\mathbf{B}(\boldsymbol{w}^{\text{sum}}-\boldsymbol{x}^{\text{sum}})\right\|_p^p$, we upper bound the magnitude of each coordinate i of $\mathbf{B}(\boldsymbol{w}^{\text{sum}}-\boldsymbol{x}^{\text{sum}})\in\mathbb{R}^m$. Suppose that constraint $C_i=\{\ell_{i,1},\ell_{i,2},\ell_{i,3}\}$ for some literals $\ell_{i,1},\ell_{i,2},\ell_{i,3}$. Then

$$(\mathbf{B}(\boldsymbol{w}^{\text{sum}}-\boldsymbol{x}^{\text{sum}}))_i = \sum_{j=1}^3 \text{sgn}(\ell_{i,j})\cdot(w^{\text{sum}}_{\text{var}(\ell_{i,j})} - x^{\text{sum}}_{\text{var}(\ell_{i,j})}) . \tag{15}$$

By Lemma 11, Item 3, we have that $|\mathbf{1}^T(\boldsymbol{w}_j - \boldsymbol{x}_j)| = |w_j^{\text{sum}} - x_j^{\text{sum}}| \leq 2$ for all $j \in [n]$. So, the expression in Equation (15) always has magnitude at most 6 by triangle inequality and has magnitude at most 4 as long as the signs of its summands are not all the same (i.e., as long as $\text{sgndiff}(\boldsymbol{b}_i, \boldsymbol{w}^{\text{sum}} - \boldsymbol{x}^{\text{sum}})$ holds, where $\boldsymbol{b}_i$ is the ith row of $\mathbf{B}$). We show that this is the case if ψ satisfies constraint C_i. Assume without loss of generality that $w^{\text{sum}}_{\text{var}(\ell_{i,j})} \neq x^{\text{sum}}_{\text{var}(\ell_{i,j})}$ for $j \in [3]$ (since otherwise the expression clearly has magnitude at most 4). By Lemma 11, Item 2 and our choice of the sign $1 - 2\psi(v_j)$ when applying Lemma 11 to $\boldsymbol{w}_j$, we have that for all $j \in [n]$, $(1 - 2\psi(v_j)) \cdot (w_j^{\text{sum}} - x_j^{\text{sum}}) \geq 0$, which implies that $(1 - 2\psi(v_j)) = \text{sgn}(w_j^{\text{sum}} - x_j^{\text{sum}})$. So, for $j \in [3]$,

$$\text{sgn}(\ell_{i,j}) \cdot \text{sgn}(w^{\text{sum}}_{\text{var}(\ell_{i,j})} - x^{\text{sum}}_{\text{var}(\ell_{i,j})}) = \text{sgn}(\ell_{i,j}) \cdot (1 - 2\psi(v_j)) = \begin{cases} -1 & \text{if } \psi \text{ satisfies literal } \ell_{i,j} \ , \\ 1 & \text{otherwise} \ . \end{cases} \tag{16}$$

By definition, a NAE-E3-SAT constraint C_i is satisfied if and only if at least one of its literals is satisfied and at least one is not satisfied. So, if ψ satisfies C_i, $\text{sgn}(\ell_{i,j}) \cdot \text{sgn}(w^{\text{sum}}_{\text{var}(\ell_{i,j})} - x^{\text{sum}}_{\text{var}(\ell_{i,j})})$ is equal to 1 for some $j \in [3]$ and is equal to -1 for some $j \in [3]$. It follows that the expression in Equation (15) has magnitude at most 4. Because $\text{val}(\phi) \geq \varepsilon$ by assumption,

$$\|(1/3) \cdot \mathbf{B}(\boldsymbol{w}^{\text{sum}} - \boldsymbol{x}^{\text{sum}})\|_p^p \leq \frac{1}{3^p} \cdot (\varepsilon m \cdot 4^p + (1 - \varepsilon) m \cdot 6^p) = (\varepsilon \cdot (4/3)^p + (1 - \varepsilon) \cdot 2^p) \cdot m \ . \tag{17}$$

Combining the upper bounds of Equations (14) and (17) to the terms in Equation (10) (with $\boldsymbol{y} := \boldsymbol{w} - \boldsymbol{x}$) implies Equation (13), as needed. ◀

3.4 Soundness Analysis

We now give two lemmas for analyzing the soundness of our reduction. The first lemma establishes that vectors of the form $\mathbf{A}\boldsymbol{x}$ where $\boldsymbol{x} \in \{0,1\}^{3n}$ are closest vectors in $\mathcal{L}(\mathbf{A})$ to $\mathbf{A} \cdot (1/2 \cdot \mathbf{1})$.

▶ **Lemma 15.** *Let $p \in [1, \infty]$, let ϕ be an* NAE-E3-SAT *instance, and let $\mathbf{A} := \mathbf{A}(\phi)$ be as defined in Equation* (8)*. Then for every $\boldsymbol{y} \in \mathbb{Z}^{3n}$, there exists $\boldsymbol{x} := (\boldsymbol{x}', \boldsymbol{x}', \boldsymbol{x}') \in \{0,1\}^{3n}$ with $\boldsymbol{x}' \in \{0,1\}^n$ such that $\|\mathbf{A}(1/2 \cdot \mathbf{1} - \boldsymbol{x})\|_p \leq \|\mathbf{A}(1/2 \cdot \mathbf{1} - \boldsymbol{y})\|_p$.*

Proof. Let $\boldsymbol{y}^1, \boldsymbol{y}^2, \boldsymbol{y}^3 \in \mathbb{Z}^n$ be such that $\boldsymbol{y} = (\boldsymbol{y}^1, \boldsymbol{y}^2, \boldsymbol{y}^3) \in \mathbb{Z}^{3n}$, let $\boldsymbol{y}^{\text{sum}} := \boldsymbol{y}^1 + \boldsymbol{y}^2 + \boldsymbol{y}^3 \in \mathbb{Z}^n$, and for each $j \in [n]$, let $\boldsymbol{y}_j := (y_j^1, y_j^2, y_j^3)$. For each $j \in [n]$, define

$$x'_j := \begin{cases} 0 & \text{if } y_j^{\text{sum}} \leq 1 \ , \\ 1 & \text{if } y_j^{\text{sum}} \geq 2 \ . \end{cases} \tag{18}$$

Furthermore, define $\boldsymbol{x}' := (x'_1, \ldots, x'_n)^T \in \{0,1\}^n$, $\boldsymbol{x} := (\boldsymbol{x}', \boldsymbol{x}', \boldsymbol{x}') \in \{0,1\}^{3n}$, and for each $j \in [n]$, $\boldsymbol{x}_j := (x'_j, x'_j, x'_j) \in \{0,1\}^3$.

We continue the proof by upper and lower bounding the ith terms in the sum in Equation (11) for $\|\mathbf{A}(1/2 \cdot \mathbf{1} - \boldsymbol{x})\|_p^p$ and for $\|\mathbf{A}(1/2 \cdot \mathbf{1} - \boldsymbol{y})\|_p^p$, respectively. Specifically, we show that the value of the ith term in the former sum is at most the value of the ith term in the latter sum for every $i \in [m]$. By the definition of $\boldsymbol{x}$ in terms of $\boldsymbol{y}$, these ith terms are the same if $\boldsymbol{y}_j \in \{\mathbf{0}, \mathbf{1}\}$ for all $j \in \text{vars}(C_i)$, where C_i is the ith constraint of ϕ. So, we assume without loss of generality that this is not the case, i.e., that for every $i \in [m]$ there exists $j \in \text{var}(C_i)$ such that $\boldsymbol{y}_j \in \mathbb{Z}^3 \setminus \{\mathbf{0}, \mathbf{1}\}$.

We begin by upper bounding the terms in $\|\mathbf{A}(1/2\cdot\mathbf{1}-\boldsymbol{x})\|_p^p$. Fix some $i \in [m]$ corresponding to a term of Equation (11). Because $\boldsymbol{x}_j \in \{\mathbf{0},\mathbf{1}\}$ for all $j \in [n]$ by definition, by Lemma 12, we have the equality

$$\sum_{j\in\mathrm{vars}(C_i)} \|\mathbf{G}(1/2\cdot\mathbf{1}-\boldsymbol{x}_j)\|_p^p = \sum_{j\in\mathrm{vars}(C_i)} 3/2^p = 3\cdot(3/2^p) = 9/2^p \ . \tag{19}$$

Let $\boldsymbol{b}_i$ be the ith row of the matrix $\mathbf{B}$ in Equation (6), as in Equation (11). Again using the fact that each $\boldsymbol{x}_j \in \{\mathbf{0},\mathbf{1}\}$, we have

$$|\langle \boldsymbol{b}_i, (1/2)\cdot\mathbf{1}-\boldsymbol{x}'\rangle| = \Big|\sum_{j\in\mathrm{vars}(C_i)} B_{i,j}\cdot(1/2-x'_j)\Big| = \begin{cases} 1/2 & \text{if } \mathrm{sgndiff}(\boldsymbol{b}_i, 1/2\cdot\mathbf{1}-\boldsymbol{x}') \ , \\ 3/2 & \text{otherwise} \ . \end{cases} \tag{20}$$

Combining Equations (19) and (20), we have

$$|\langle \boldsymbol{b}_i, (1/2)\cdot\mathbf{1}-\boldsymbol{x}'\rangle|^p + \sum_{j\in C_i} \|\mathbf{G}(1/2\cdot\mathbf{1}-\boldsymbol{x}_j)'\|_p^p = \begin{cases} 10/2^p & \text{if } \mathrm{sgndiff}(\boldsymbol{b}_i, 1/2\cdot\mathbf{1}-\boldsymbol{x}') \ , \\ (9+3^p)/2^p & \text{otherwise} \ . \end{cases} \tag{21}$$

We now lower bound $\|\mathbf{A}(1/2\cdot\mathbf{1}-\boldsymbol{y})\|_p^p$. We have that

$$\sum_{j\in\mathrm{vars}(C_i)} \|\mathbf{G}(1/2\cdot\mathbf{1}-\boldsymbol{y}_j)\|_p^p \geq (3/2)^p + 8/2^p \ . \tag{22}$$

Indeed, by Lemma 12, the terms for which $\boldsymbol{y}_j \in \{\mathbf{0},\mathbf{1}\}$ are equal to $3/2^p$ and the terms in the sum for which $\boldsymbol{y}_j \notin \{\mathbf{0},\mathbf{1}\}$ must have value at least $(3/2)^p + 2/2^p$. And, since we are only considering constraints C_i for which there exists j such that $\boldsymbol{y}_j \notin \{\mathbf{0},\mathbf{1}\}$, at least one term must have value at least $(3/2)^p + 2/2^p$.

We continue by giving a lower bound for $|(1/3)\cdot\boldsymbol{b}_i(3/2\cdot\mathbf{1}-\boldsymbol{y}^{\mathrm{sum}})|$. Here, we observe that $(1/2)-(y_j^{\mathrm{sum}}/3) \in \{1/6,3/6,5/6\}+\mathbb{Z}$. This implies that

$$\begin{aligned} \langle \boldsymbol{b}_i, (1/2)\cdot\mathbf{1}-\boldsymbol{y}^{\mathrm{sum}}/3\rangle &= \Big|\sum_{j\in\mathrm{vars}(C_i)} B_{i,j}\cdot(1/2-y_j^{\mathrm{sum}}/3)\Big| \\ &\geq \begin{cases} 1/6 & \text{if } \mathrm{sgndiff}(\boldsymbol{b}_i, 1/2\cdot\mathbf{1}-\boldsymbol{y}^{\mathrm{sum}}/3) \\ 1/2 & \text{otherwise} \end{cases} , \end{aligned} \tag{23}$$

where the inequality follows by noting that because all three summands in the middle expression are elements of $\{1/6,3/6,5/6\}+\mathbb{Z}$. If all signs of the summands are the same, one obtains the lower bound $3/6 = 1/2$, whereas if the signs of the summands differ, one only has the lower bound of $1/6$.

By combining Equations (22) and (23), we have the lower bound

$$\begin{aligned} |\langle \boldsymbol{b}_i, (1/2)\cdot\mathbf{1}-\boldsymbol{y}^{\mathrm{sum}}/3\rangle|^p + &\sum_{j\in\mathrm{vars}(C_i)} \|\mathbf{G}(1/2\cdot\mathbf{1}-\boldsymbol{y}_j)\|_p^p \\ &\geq \begin{cases} 1/6^p + (3/2)^p + 8/2^p & \text{if } \mathrm{sgndiff}(\boldsymbol{b}_i, 1/2\cdot\mathbf{1}-\boldsymbol{y}^{\mathrm{sum}}/3) \\ (9+3^p)/2^p & \text{otherwise} \end{cases} . \end{aligned} \tag{24}$$

Finally, by definition of $\boldsymbol{x}'$ (Equation (18)), we have that for all $j \in [n]$, $\mathrm{sgn}(x'_j - 1/2) = \mathrm{sgn}(y_j^{\mathrm{sum}}/3 - 1/2)$. Then, it follows that for all $i \in [m]$ that $\mathrm{sgndiff}(\boldsymbol{b}_i, 1/2 \cdot \mathbf{1} - \boldsymbol{x}')$ if and only if $\mathrm{sgndiff}(\boldsymbol{b}_i, 1/2 \cdot \mathbf{1} - \boldsymbol{y}^{\mathrm{sum}}/3)$. By noting that for all $p \geq 1$, $2/2^p \leq 1/6^p + (3/2)^p$, we combine Equations (21) and (24) to obtain that for all $i \in [m]$,

$$\begin{aligned} &|\langle \boldsymbol{b}_i, (1/2) \cdot \mathbf{1} - \boldsymbol{x}' \rangle|^p + \sum_{j \in \mathrm{vars}(C_i)} \|\mathbf{G}(1/2 \cdot \mathbf{1} - \boldsymbol{x}_j)\|_p^p \\ &\leq |\langle \boldsymbol{b}_i, (1/2) \cdot \mathbf{1} - \boldsymbol{y}^{\mathrm{sum}}/3 \rangle|^p + \sum_{j \in \mathrm{vars}(C_i)} \|\mathbf{G}(1/2 \cdot \mathbf{1} - \boldsymbol{y}_j)\|_p^p \ , \end{aligned}$$

where we note that the left- and right-hand sides of the inequality are equal to the ith term in the sum in Equation (11). The result follows. ◀

The following lemma shows soundness for our reduction. Specifically, it shows how to construct an assignment satisfying a large fraction of constraints in an NAE-E3-SAT formula ϕ from a close vector $\mathbf{A}\boldsymbol{x}$ for $\boldsymbol{x} \in \{0,1\}^{3n}$ to $\mathbf{A}(1/2 \cdot \mathbf{1}_{3n})$, where $\mathbf{A} = \mathbf{A}_p(\phi)$.

▶ **Lemma 16.** *Let $p \in [1, \infty)$, let ϕ be an* NAE-E3-SAT *instance with n variables and m constraints, let $\mathbf{A} := \mathbf{A}_p(\phi)$ be as defined in Equation* (8)*, and let $\delta \in [0,1]$. Suppose that there exists some vector $\boldsymbol{x} := (\boldsymbol{x}^1, \boldsymbol{x}^2, \boldsymbol{x}^3) \in \mathbb{Z}^{3n}$ such that*

$$\|\mathbf{A}((1/2) \cdot \mathbf{1} - \boldsymbol{x})\|_p^p \leq \delta m \cdot (10/2^p) + (1 - \delta)m \cdot (9/2^p + (3/2)^p) \ . \tag{25}$$

Then $\mathrm{val}(\phi) \geq \delta$.

Proof. Suppose that there exists $\boldsymbol{x} \in \mathbb{Z}^{3n}$ satisfying Equation (25). Then by Lemma 15, we may assume without loss of generality that $\boldsymbol{x} = (\boldsymbol{x}', \boldsymbol{x}', \boldsymbol{x}')$ for $\boldsymbol{x}' \in \{0,1\}^n$ and let $\boldsymbol{x}_j := (x'_j, x'_j, x'_j)$ for each $j \in [n]$. Therefore, for each $i \in [m]$, both Equations (19) and (20) hold for $\boldsymbol{x}$.

First, we obtain the bound

$$\begin{aligned} \frac{1}{3^p} \cdot \|\mathbf{B}((3/2) \cdot \mathbf{1} - \boldsymbol{x}^{\mathrm{sum}})\|_p^p &= \|\mathbf{A}((1/2) \cdot \mathbf{1} - \boldsymbol{x})\|_p^p - \sum_{j=1}^{n} \deg(v_j) \cdot \|\mathbf{G}((1/2) \cdot \mathbf{1} - \boldsymbol{x}_j)\|_p^p \\ &= \|\mathbf{A}((1/2) \cdot \mathbf{1} - \boldsymbol{x})\|_p^p - m \cdot (9/2^p) \\ &\leq \delta m \cdot (1/2^p) + (1 - \delta)m \cdot (3/2)^p \ , \end{aligned}$$

where the first equality uses Equation (10) with $\boldsymbol{y} := (1/2) \cdot \mathbf{1} - \boldsymbol{x}$, the second equality uses Equation (19), and the inequality is by assumption (Equation (25)).

It then follows from Equation (20) that at least an δ fraction of indices $i \in [m]$ satisfy $\mathrm{sgndiff}(\boldsymbol{b}_i, 1/2 \cdot \mathbf{1} - \boldsymbol{x}')$. Define an assignment ψ to the variables $v_1, \ldots, v_n$ of ϕ by $\psi(v_j) = x'_j$. We claim that if $\mathrm{sgndiff}(\boldsymbol{b}_i, 1/2 \cdot \mathbf{1} - \boldsymbol{x}')$ then ψ satisfies constraint C_i, which combined with the fact that at least an δ fraction of indices $i \in [m]$ satisfy $\mathrm{sgndiff}(\boldsymbol{b}_i, 1/2 \cdot \mathbf{1} - \boldsymbol{x}')$, would imply the lemma.

It remains to prove the claim. Observe that if constraint $C_i = \{\ell_{i,1}, \ell_{i,2}, \ell_{i,3}\}$ for literals $\ell_{i,1}, \ell_{i,2}, \ell_{i,3}$ then, as in Equation (15),

$$(\mathbf{B}(1/2 \cdot \mathbf{1} - \boldsymbol{x}'))_i = \sum_{j=1}^{3} \mathrm{sgn}(\ell_{i,j}) \cdot (1/2 - x_{\mathrm{var}(\ell_{i,j})}) \ .$$

Furthermore, as in Equation (16), for $j \in [3]$,

$$\mathrm{sgn}(\ell_{i,j}) \cdot \mathrm{sgn}(1/2 - x_{\mathrm{var}(\ell_{i,j})}) = \begin{cases} -1 & \text{if } \psi \text{ satisfies literal } \ell_{i,j} \ , \\ 1 & \text{otherwise} \ . \end{cases}$$

The predicate $\mathrm{sgndiff}(\boldsymbol{b}_i, 1/2 \cdot \mathbf{1} - \boldsymbol{x}')$ holds exactly when $\mathrm{sgn}(\ell_{i,j}) \cdot (1/2 - x_{\mathrm{var}(\ell_{i,j})}) = 1$ for some $j \in [3]$ and $\mathrm{sgn}(\ell_{i,j}) \cdot (1/2 - x_{\mathrm{var}(\ell_{i,j})}) = -1$ for some $j \in [3]$, meaning that ψ satisfies at least one literal in C_i and does not satisfy at least one literal in C_i. This in turn implies that ψ satisfies C_i, as needed. ◀

3.5 The Main Theorem

We now state and prove the main technical theorem of the paper.

▶ **Theorem 17.** *Let $\delta, \varepsilon \in \mathbb{R}$ be such that $0 < \delta \leq \varepsilon \leq 1$ and let $p \in [1, \infty)$. There is a polynomial-time mapping reduction from (δ, ε)-NAE-E3-SAT to γ-BinGapCRP$_p$ for*

$$\gamma = \gamma(\delta, \varepsilon, p) := \left(\frac{\delta \cdot 10/2^p + (1-\delta) \cdot ((3/2)^p + 9/2^p)}{\varepsilon \cdot (4/3)^p + (1-\varepsilon)2^p + 3(2 + (4/3)^p)} \right)^{1/p} . \tag{26}$$

Proof. Let ϕ be the input instance of (δ, ε)-NAE-E3-SAT. Assume that ϕ has m constraints. The reduction outputs the BinGapCRP$_p$ instance consisting of the matrix $\mathbf{A} = \mathbf{A}_p(\phi)$ defined in Equation (8) and the distance threshold $r := (\varepsilon m \cdot (4/3)^p + (1-\varepsilon) m \cdot 2^p + 3m(2 + (4/3)^p))^{1/p}$ defined in Equation (9).

It is clear that the reduction runs in polynomial time, and it remains to show correctness. We start with completeness. If ϕ is a YES instance of (δ, ε)-NAE-E3-SAT, then $\mathrm{lindisc}_p(\mathbf{A}) \leq r$ by Lemma 14 and so the output is a YES instance of BinGapCRP$_p$.

For soundness, we show the contrapositive. Suppose that

$$\mu_p(\mathcal{L}(\mathbf{A})) \leq \gamma r = (\delta m \cdot 10/2^p + (1-\delta) m \cdot ((3/2)^p + 9/2^p))^{1/p} .$$

Then there exists $\boldsymbol{y} \in \mathbb{Z}^{3n}$ such that $\|\mathbf{A}((1/2 \cdot \mathbf{1}) - \boldsymbol{y})\|_p \leq \gamma r$, and so $\mathrm{val}(\phi) \geq \delta$ by Lemma 16. The result follows. ◀

We conclude by obtaining Theorem 1, which asserts that γ-BinGapCRP$_p$ is NP-hard for any constant $1 \leq \gamma < \gamma(p)$, as a corollary of Theorem 17.

Proof of Theorem 1. Let $\gamma(p) := \gamma(15/16, 1, p)$, where $\gamma(\cdot, \cdot, \cdot)$ is as defined in Equation (26). Then

$$\begin{aligned} \gamma(p) &= \left(\frac{(15/16) \cdot 10/2^p + (1/16) \cdot ((3/2)^p + 9/2^p)}{(4/3)^p + 3(2 + (4/3)^p)} \right)^{1/p} \\ &= \left(\frac{9^p + 159 \cdot 3^p}{8^{p+2} + 96 \cdot 6^p} \right)^{1/p} , \end{aligned}$$

By Theorem 10, $(15/16 + \varepsilon', 1)$-NAE-E3-SAT is NP-hard for every constant $\varepsilon' > 0$. So, by Theorem 17, $\gamma(15/16 + \varepsilon', 1, p)$-BinGapCRP is NP-hard for every such ε'. The theorem follows by noting that for every constant $\varepsilon > 0$ there exists a constant $\varepsilon' > 0$ such that $\gamma(15/16 + \varepsilon', 1, p) \geq \gamma(p) - \varepsilon$. ◀

4 Π_2-hardness of approximating BinaryGapCRP in the ℓ_∞ norm

In this section, we show that the Π_2-hardness of approximation result for LinDisc$_\infty$ given in Manurangsi [29] can be extended to BinGapCRP$_\infty$. We first give the definition of $\forall\exists$-NAE-E3-SAT, the Π_2-hard problem that we reduce to approximate BinGapCRP. The problem $\forall\exists$-NAE-3-SAT (in which constraints may have either two or three literals) was originally shown to be Π_2-hard in [12], and (a very special case of) $\forall\exists$-NAE-E3-SAT was shown to be Π_2-hard in [10, Theorem 4.7].

▶ **Definition 18** (∀∃-NAE-E3-SAT). *The* ∀∃-NAE-E3-SAT *problem is the decision problem where, on input a* NAE-E3-SAT *formula $\phi(V_A, V_E)$ defined over the disjoint union $V_A \sqcup V_E$ of variable sets V_A and V_E, the goal is to distinguish between the following two cases.*

- *(YES instance.) For every assignment $\psi_A : V_A \to \{0,1\}$, there exists an assignment $\psi_E : V_E \to \{0,1\}$ such that (ψ_A, ψ_E) satisfies ϕ.*
- *(NO instance.) There exists an assignment $\psi_A : V_A \to \{0,1\}$ such that for all assignments $\psi_E : V_E \to \{0,1\}$), (ψ_A, ψ_E) does not satisfy ϕ.*

Let $\phi(V_A, V_E)$ be a NAE-E3-SAT formula with $n' = |V_A|$ and $n - n' = |V_E|$. We define the matrix $\mathbf{A}' = \mathbf{A}'(\phi) \in \mathbb{R}^{(m+3n+2n')\times(3n+n')}$ (originally defined in [29]) as

$$\mathbf{A}' := \left(\begin{array}{cccccc|c} \multicolumn{6}{c|}{\mathbf{A}_\infty(\phi)} & \mathbf{0}_{n'\times n'} \\ \hline \frac{2}{3}\mathbf{I}_{n'} & \mathbf{0}_{n'\times(n-n')} & \frac{2}{3}\mathbf{I}_{n'} & \mathbf{0}_{n'\times(n-n')} & \frac{2}{3}\mathbf{I}_{n'} & \mathbf{0}_{n'\times(n-n')} & -2\mathbf{I}_{n'} \\ \hline \multicolumn{6}{c|}{\mathbf{0}_{n'\times 3n}} & \frac{8}{3}\mathbf{I}_{n'} \end{array}\right), \tag{27}$$

where $\mathbf{A}_\infty(\phi)$ is as defined in Equation (8) and where we have used the notation $\mathbf{0}_{m\times n}$ to denote an $m \times n$ matrix of 0s.

We next give a simple expression for $\|\mathbf{A}'(\boldsymbol{y}')\|_\infty$ for a vector $\boldsymbol{y}'$ in terms of the blocks of $\mathbf{A}'$.

▶ **Lemma 19** ([29, Equation 4]). *Let $\phi(V_A, V_E)$ be an* ∀∃-NAE-E3-SAT *instance with n' universal variables (variables in V_A) and $n - n'$ existential variables (variables in V_E), and for each $i \in [3]$, let $\boldsymbol{y}^i \in \mathbb{R}^n$, and let $\boldsymbol{y}^* \in \mathbb{R}^{n'}$. Furthermore, let $\boldsymbol{y}' := (\boldsymbol{y}^1, \boldsymbol{y}^2, \boldsymbol{y}^3, \boldsymbol{y}^*) = (\boldsymbol{y}, \boldsymbol{y}^*) \in \mathbb{R}^{3n+n'}$. Finally, let $\boldsymbol{y}^{sum} = \boldsymbol{y}^1 + \boldsymbol{y}^2 + \boldsymbol{y}^3$. Then $\mathbf{A}' = \mathbf{A}'(\phi)$ satisfies*

$$\|\mathbf{A}'\boldsymbol{y}'\|_\infty = \max\{\|\mathbf{A}\boldsymbol{y}\|_\infty, \max_{i\in[n']}|(2/3)\cdot y_i^{sum} - 2y_i^*|, (8/3)\cdot\max_{i\in[n']}|y_i^*|\} \tag{28}$$

Proof. The claim follows by inspection of the blocks of $\mathbf{A}'\boldsymbol{y}'$. ◀

We next state the completeness and soundness results from the reduction in [29] for showing Π_2-hardness of $\mathrm{lindisc}_\infty(A)$.

▶ **Lemma 20** (Completeness for Π_2-hardness reduction, [29]). *Let ϕ be an* ∀∃-NAE-E3-SAT *instance, and $\mathbf{A}' := \mathbf{A}'(\phi)$ as defined in Equation* (27)*. Suppose that ϕ is a YES instance. Then $\mathrm{lindisc}_\infty(\mathbf{A}') \le 4/3$.*

Proof. The claim follows from the completeness argument in [29, Proof of Lemma 5]. ◀

▶ **Lemma 21** (Soundness for Π_2-hardness reduction, [29]). *Let $\phi(V_A, V_E)$ be an instance of* ∀∃-NAE-E3-SAT *with $n' := |V_A|$ universally quantified variables and $n := |V_A| + |V_E|$ variables, and let $\mathbf{A}' = \mathbf{A}'(\phi)$ be as defined in Equation* (27)*. Suppose that for every $\boldsymbol{w}' := (\boldsymbol{w}, \boldsymbol{w}^*)$ for $\boldsymbol{w} := 1/2 \cdot \mathbf{1} \in \mathbb{R}^{3n}$ and $\boldsymbol{w}^* \in \{1/3, 2/3\}^n$ there exists a vector $\boldsymbol{x}' \in \{0,1\}^{3n+n'}$ such that $\|\mathbf{A}'(\boldsymbol{w}' - \boldsymbol{x}')\|_\infty < 3/2$. Then ϕ is a YES instance of* ∀∃-NAE-E3-SAT*.*

Proof. The claim follows from the soundness argument in [29, Proof of Lemma 5]. ◀

We now state a lemma that guarantees that if a vector $\boldsymbol{w}'$ of the form in Lemma 21 and $\boldsymbol{x}' \in \mathbb{Z}^{3n+n'}$ satisfy $\|\mathbf{A}'(\boldsymbol{w}' - \boldsymbol{x}')\|_\infty < 3/2$, then it must be the case that $\boldsymbol{x}' \in \{0,1\}^{3n+n'}$.

▶ **Lemma 22.** *Let $\phi(V_A, V_E)$ be an* ∀∃-NAE-E3-SAT *instance with $n' := |V_A|$ universally quantified variables and $n := |V_A| + |V_E|$ variables, and let $\mathbf{A}' = \mathbf{A}'(\phi)$ be as defined in Equation* (27)*. Furthermore, let $\boldsymbol{w} := 1/2 \cdot \mathbf{1} \in \mathbb{R}^{3n}$, let $\boldsymbol{w}^* \in \{1/3, 2/3\}^{n'}$, let $\boldsymbol{x} \in \mathbb{Z}^{3n}$, and let $\boldsymbol{x}^* \in \mathbb{Z}^{n'}$. Let $\boldsymbol{w}' := (\boldsymbol{w}, \boldsymbol{w}^*) \in \mathbb{R}^{3n+n'}$ and let $\boldsymbol{x}' := (\boldsymbol{x}, \boldsymbol{x}^*) \in \mathbb{Z}^{3n+n'}$. Assume that $\|\mathbf{A}'(\boldsymbol{w}' - \boldsymbol{x}')\|_\infty < 3/2$. Then $\boldsymbol{x}' \in \{0,1\}^{3n+n'}$.*

Proof. We have that

$$\max_{j\in[n]}\|\mathbf{G}(\boldsymbol{w}_j-\boldsymbol{x}_j)\|_\infty \le \|\mathbf{A}(\boldsymbol{w}-\boldsymbol{x})\|_\infty \le \|\mathbf{A}'(\boldsymbol{w}'-\boldsymbol{x}')\|_\infty < 3/2\ , \tag{29}$$

where the first inequality follows from the definition of $\mathbf{A}$ (Equation (8)), the second inequality follows from Lemma 19, and the third inequality is an assumption. Let $\boldsymbol{x}^1,\boldsymbol{x}^2,\boldsymbol{x}^3\in\mathbb{Z}^n$ be such that $\boldsymbol{x}:=(\boldsymbol{x}^2,\boldsymbol{x}^2,\boldsymbol{x}^3)$, and for $j\in[n]$, let $\boldsymbol{x}_j:=(x_j^1,x_j^2,x_j^3)\in\mathbb{Z}^3$. By Equation (29) we then have that for all $j\in[n]$, $\|\mathbf{G}(1/2\cdot\mathbf{1}-\boldsymbol{x}_j)\|_\infty<3/2$. Therefore, by Lemma 12 we have have that for all $j\in[n]$, $\boldsymbol{x}_j\in\{\mathbf{0},\mathbf{1}\}$. In particular, $\boldsymbol{x}\in\{0,1\}^{3n}$.

Again, by setting $\boldsymbol{y}':=\boldsymbol{w}'-\boldsymbol{x}'$ in Lemma 19, we have that $(8/3)\cdot\max_{i\in[n']}|w_i^*-x_i^*|<3/2$, or equivalently, that for all $i\in[n']$, $|w_i^*-x_i^*|<9/16<2/3$. It follows that if $w_i^*=1/3$ then $x_i^*=0$, and if $w_i^*=2/3$ then $x_i^*=1$. In particular, $\boldsymbol{x}^*\in\{0,1\}^{n'}$, as needed. ◀

Finally, we obtain the following theorem that adapts [29, Lemma 5] to the BinGapCRP$_\infty$ problem.

▶ **Theorem 23.** *For any constant $\varepsilon>0$, there is a polynomial-time mapping reduction from* ∀∃-NAE-E3-SAT *to* $(9/8-\varepsilon)$-BinGapCRP$_\infty$.

Proof. Let ϕ be an instance of the ∀∃-NAE-E3-SAT problem. The reduction outputs the BinGapCRP instance with matrix $\mathbf{A}':=\mathbf{A}'(\phi)$, as defined in Equation (27), and distance threshold $r:=4/3$. The reduction is efficient by inspection. If ϕ is a YES instance, then $\mathrm{lindisc}_\infty(\mathbf{A}')\le 4/3$ by Lemma 20.

On the other hand, if $\mu_\infty(\mathcal{L}(\mathbf{A}'))<3/2$, then by definition of the covering radius and Lemma 22 we have that for all $\boldsymbol{w}':=(\boldsymbol{w},\boldsymbol{w}^*)\in\mathbb{R}^{3n+n'}$ with $\boldsymbol{w}:=1/2\cdot\mathbf{1}^{3n}$ and $\boldsymbol{w}^*\in\{1/3,2/3\}^{n'}$, there exists $\boldsymbol{x}'\in\{0,1\}^{3n+n'}$ such that $\|\mathbf{A}'(\boldsymbol{w}'-\boldsymbol{x}')\|_\infty<3/2$. Therefore, ϕ is a YES instance by Lemma 21. The result follows. ◀

The proof of Theorem 2 now follows easily.

Proof of Theorem 2. The result follows from the Π_2-hardness of ∀∃-NAE-E3-SAT (shown in [10, Theorem 4.7]) and the reduction in Theorem 23. ◀

References

1 Divesh Aggarwal and Noah Stephens-Davidowitz. (gap/s)eth hardness of SVP. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 228–238. ACM, 2018. doi:10.1145/3188745.3188840. 5

2 Miklós Ajtai. The shortest vector problem in L_2 is *NP*-hard for randomized reductions (extended abstract). In Jeffrey Scott Vitter, editor, *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 10–19. ACM, 1998. doi:10.1145/276698.276705. 2

3 Huck Bennett. The complexity of the Shortest Vector Problem. *SIGACT News*, 54(1):37–61, 2023. doi:10.1145/3586165.3586172. 3

4 Huck Bennett and Chris Peikert. Hardness of bounded distance decoding on lattices in p norms. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, Saarbrücken, Germany (Virtual Conference), July 28-31, 2020*, volume 169 of *LIPIcs*, pages 36:1–36:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CCC.2020.36. 5

5 Huck Bennett, Chris Peikert, and Yi Tang. Improved hardness of BDD and SVP under gap-(s)eth. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, Berkeley, CA, USA, January 31 - February 3, 2022*, volume 215 of *LIPIcs*, pages 19:1–19:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ITCS.2022.19. 5

6 Joshua Brakensiek, Neng Huang, Aaron Potechin, and Uri Zwick. On the mysteries of MAX NAE-SAT. *SIAM J. Discret. Math.*, 39(1):267–313, 2025. doi:10.1137/23m1591578. 7

7 William Chen, Anand Srivastav, and Giancarlo Travaglini. *A Panorama of Discrepancy Theory*. Springer International Publishing, 2014. doi:10.1007/978-3-319-04696-9. 5

8 Daniel Dadush. On approximating the covering radius and finding dense lattice subspaces. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1021–1026. ACM, 2019. doi:10.1145/3313276.3316397. 1

9 Irit Dinur, Elchanan Mossel, and Oded Regev. Conditional hardness for approximate coloring. *SIAM J. Comput.*, 39(3):843–873, 2009. Preliminary version in STOC 2006. doi:10.1137/07068062X. 4

10 Janosch Döcker, Britta Dorn, Simone Linz, and Charles Semple. Placing quantified variants of 3-sat and not-all-equal 3-sat in the polynomial hierarchy. *Theor. Comput. Sci.*, 822:72–91, 2020. doi:10.1016/J.TCS.2020.04.003. 17, 19

11 Friedrich Eisenbrand, Dömötör Pálvölgyi, and Thomas Rothvoß. Bin packing via discrepancy of permutations. *ACM Trans. Algorithms*, 9(3):24:1–24:15, 2013. doi:10.1145/2483699.2483704. 5

12 Thomas Eiter and Georg Gottlob. Note on the complexity of some eigenvector problems. Technical Report CD-TR 95/89, Christian Doppler Laboratory for Expert Systems, TU Vienna, 1995. 17

13 Aparna Gupte and Vinod Vaikuntanathan. The fine-grained hardness of sparse linear regression. *CoRR*, abs/2106.03131, 2021. arXiv:2106.03131. 3

14 Venkatesan Guruswami. Inapproximability results for set splitting and satisfiabilityproblems with no mixed clauses. *Algorithmica*, 38(3):451–469, March 2004. doi:10.1007/s00453-003-1072-z. 7, 10

15 Venkatesan Guruswami, Daniele Micciancio, and Oded Regev. The complexity of the covering radius problem on lattices and codes. In *19th Annual IEEE Conference on Computational Complexity (CCC 2004), 21-24 June 2004, Amherst, MA, USA*, pages 161–173. IEEE Computer Society, 2004. doi:10.1109/CCC.2004.1313831. 2, 3, 4, 7

16 Isaac M Hair and Amit Sahai. SVP_p is deterministically NP-hard for all $p > 2$, even to approximate within a factor of $2^{\log^{1-\varepsilon} n}$. Cryptology ePrint Archive, Paper 2025/2181, 2025. URL: https://eprint.iacr.org/2025/2181. 2, 5

17 Isaac M Hair and Amit Sahai. Deterministic hardness of approximation for SVP in all finite ℓ_p norms, 2026. doi:10.48550/arXiv.2604.01451. 2, 5

18 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001. doi:10.1145/502090.502098. 7, 10

19 Ishay Haviv, Vadim Lyubashevsky, and Oded Regev. A note on the distribution of the distance from a lattice. *Discret. Comput. Geom.*, 41(1):162–176, 2009. doi:10.1007/S00454-008-9123-5. 4

20 Ishay Haviv and Oded Regev. Hardness of the covering radius problem on lattices. *Chic. J. Theor. Comput. Sci.*, 2012, 2012. URL: http://cjtcs.cs.uchicago.edu/articles/2012/4/contents.html. 2, 3, 4

21 Ishay Haviv and Oded Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. *Theory Comput.*, 8(1):513–531, 2012. doi:10.4086/TOC.2012.V008A023. 2

22 Yahli Hecht and Muli Safra. Deterministic hardness of approximation of unique-svp and gapsvp in ℓ_p norms for $p > 2$, 2025. arXiv:2510.16991. 2

23 Rebecca Hoberg and Thomas Rothvoss. A logarithmic additive integrality gap for bin packing. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2616–2625. SIAM, 2017. doi:10.1137/1.9781611974782.172. 5

24 Subhash Khot. Hardness of approximating the shortest vector problem in lattices. *J. ACM*, 52(5):789–808, 2005. Preliminary version in FOCS 2004. doi:10.1145/1089023.1089027. 2, 5

25 Subhash Khot. Hardness of approximating the shortest vector problem in high l_p norms. *J. Comput. Syst. Sci.*, 72(2):206–219, 2006. Preliminary version in FOCS 2003. doi:10.1016/J.JCSS.2005.07.002. 2, 5

26 Lily Li and Aleksandar Nikolov. On the Computational Complexity of Linear Discrepancy. In Fabrizio Grandoni, Grzegorz Herman, and Peter Sanders, editors, *28th Annual European Symposium on Algorithms (ESA 2020)*, volume 173 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 69:1–69:16, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ESA.2020.69. 5

27 L. Lovász, J. Spencer, and K. Vesztergombi. Discrepancy of set-systems and matrices. *European Journal of Combinatorics*, 7(2):151–160, 1986. doi:10.1016/S0195-6698(86)80041-5. 5, 8

28 Alexander Magazinov. A proof of a conjecture by haviv, lyubashevsky and regev on the second moment of a lattice voronoi cell. *Advances in Geometry*, 20(1):117–120, January 2020. doi:10.1515/advgeom-2018-0018. 2, 4

29 Pasin Manurangsi. Linear discrepancy is Π_2-hard to approximate. *Information Processing Letters*, 172:106164, 2021. doi:10.1016/j.ipl.2021.106164. 2, 3, 5, 6, 7, 10, 11, 12, 17, 18, 19

30 Aileen M. McLoughlin. The complexity of computing the covering radius of a code. *IEEE Trans. Inf. Theory*, 30(6):800–804, 1984. doi:10.1109/TIT.1984.1056978. 3

31 Daniele Micciancio. The shortest vector in a lattice is hard to approximate to within some constant. *SIAM J. Comput.*, 30(6):2008–2035, 2000. Preliminary version in FOCS 1998. doi:10.1137/S0097539700373039. 2

32 Daniele Micciancio. Almost perfect lattices, the covering radius problem, and applications to ajtai's connection factor. *SIAM J. Comput.*, 34(1):118–169, 2004. doi:10.1137/S0097539703433511. 1, 4

33 Daniele Micciancio. Inapproximability of the shortest vector problem: Toward a deterministic reduction. *Theory Comput.*, 8(1):487–512, 2012. doi:10.4086/TOC.2012.V008A022. 2

34 S. Muthukrishnan and Aleksandar Nikolov. Optimal private halfspace counting via discrepancy. In *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '12, pages 1285–1292, New York, NY, USA, 2012. Association for Computing Machinery. doi:10.1145/2213977.2214090. 5

35 Erez Petrank. The hardness of approximation: Gap location. *computational complexity*, 4(2):133–157, June 1994. doi:10.1007/BF01202286. 7, 10

36 Oded Regev and Ricky Rosen. Lattice problems and norm embeddings. In Jon M. Kleinberg, editor, *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 447–456. ACM, 2006. doi:10.1145/1132516.1132581. 2

37 Thomas Rothvoss. Better bin packing approximations via discrepancy theory. *SIAM J. Comput.*, 45(3):930–946, 2016. doi:10.1137/140955367. 5

38 Daqing Wan. Np-hardness of svp in euclidean space, 2026. doi:10.48550/arXiv.2603.27398. 2

Improved Algorithms and Lower Bounds for Parametrized Metrical Service Systems

Junhao Gan
School of Computing and Information Systems, The University of Melbourne, Australia

Xiao Sun
School of Computing and Information Systems, The University of Melbourne, Australia

Seeun William Umboh
School of Computing and Information Systems, The University of Melbourne, Australia
ARC Training Centre in Optimisation Technologies, Integrated Methodologies, and Applications (OPTIMA), Melbourne, Australia

Abstract

We consider the parametrized setting of the classical metrical service system (MSS) problem first studied by Bubeck and Rabani (APPROX/RANDOM 2020). In this setting, the adversary is restricted to a set of m distinct request types, known to the algorithm in advance. The goal is to obtain competitive ratio bounds in terms of m. In this work, we make significant progress in understanding the landscape of parametrized MSS and resolve several open problems from Bubeck and Rabani.

Our first main result is a tight bound for parametrized MSS on weighted stars. Previously, Bubeck and Rabani gave a randomized lower bound of $\Omega(m)$ and deterministic upper bound of $O(2^m)$. We show that, surprisingly, a deterministic $O(m)$-competitive algorithm exists, matching the randomized lower bound. Our key insight is an interval covering formulation of MSS on weighted stars which enables an application of the primal-dual method.

Our second main contribution is an improved lower bound construction for parametrized MSS on hierarchically separated trees (HSTs). Bubeck and Rabani's construction gave a $\omega(1)$ lower bound when $m \geq 6$. Our improved lower bounds are tight for 2-level HSTs and also rule out $O(1)$-competitive algorithms on HSTs when the parameter $m \geq 4$. We also complement these results by giving a deterministic $O(1)$-competitive algorithm on general metrics when $m = 2$ while showing that it is impossible when $m \geq 3$.

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases online algorithms, competitive analysis, metrical service systems, metrical task systems

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.11

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2607.07098

Funding *Junhao Gan*: Supported in part by the Australian Government through the Australian Research Council ARC DP230102908.
Xiao Sun: Supported by the Australian Government through the Australian Research Council DP240101353 and by the University of Melbourne through the Melbourne Research Scholarship.
Seeun William Umboh: Supported by the Australian Government through the Australian Research Council DP240101353.

Acknowledgements We thank Christian Coester for insight into the case of $m = 3$ on general metrics.


© Junhao Gan, Xiao Sun, and Seeun William Umboh;
licensed under Creative Commons License CC-BY 4.0



1 Introduction

The *metrical task system* (MTS) problem introduced by Borodin et al. [2] is a general framework that models a wide range of online computing problems, e.g., paging [15] and k-server [13]. In this problem, one server moves between different points (also called states) in an n-point metric space to process a sequence of requests that arrive one by one; each request incurs different costs when the server is at different states. The total cost is the sum of the transition cost of moving between different states and the cost of serving the requests. An important special case is the *metrical service system* (MSS) problem [9] where every request has cost either 0 or infinity everywhere in the whole metric space. Note that paging and k-server are both MSS problems.

Since its introduction almost four decades ago, MTS has been intensively studied on both general metrics and some special classes of metrics. On general metrics, in a series of recent breakthrough results, Bubeck et al. [4] showed that the randomized competitive ratio is $O(\log^2 n)$ and Bubeck et al. [3] gave a matching *existential lower bound*[1], refuting the previous widely-believed conjecture of the existence of a logarithmic upper bound. Bubeck et al. [3] also showed a *universal lower bound*[2] of $\Omega(\log n)$, which is also tight in the sense that an $O(\log n)$ upper bound can be achieved on some special metrics, e.g. uniform metrics [2] and weighted stars [4]. While MSS is a special case of MTS, the best competitive ratio that can be achieved by both deterministic and randomized algorithms for MSS in all cases is asymptotically the same as that of MTS only up to a constant factor.

While many online problems are metrical task systems, the best bounds on their competitive ratios are much lower than those implied by the general worst-case MTS bounds. For instance, while the k-server problem in an N-point uniform metric space can be reduced to an MSS instance with $n = \binom{N}{k}$ states, the randomized competitive ratio is known to be $O(\log k)$, which is much lower than the lower bound of $\Omega(\log \binom{N}{k})$ implied by the universal lower bound for MSS with $\binom{N}{k}$ states. The reason for this discrepancy is because while there are $2^n - 1$ possible requests in a general n-state MSS, there are only N possible requests for a k-server.

The above example motivates studying the competitive ratio of MTS in terms of a parameter of the set of possible requests, instead of the number of states n. Burley and Irani [8] proposed examining request sequences drawn from a given set. For uniform metrics, they provided a deterministic algorithm with competitive ratio $O(\log n)$ times the (unknown) best possible competitive ratio. Another example that shows the power of restricting the set of possible requests is the power management problem [11] in which there are only two possible requests and the competitive ratio is $O(1)$ [1]. For MSS, one parameter that has been studied is the *width* (e.g. [10, 14, 7]), which is the maximum size of every requested set. Width is well-motivated as MSS can also be viewed as online set chasing, in the sense that the server needs to move to a state in the feasible set where the cost of the current request is 0 rather than ∞, and we only need to consider transition cost while the service cost is zero if the server stays in the feasible set at every step.

Parametrized MTS. The parametrized MTS proposed by Bubeck and Rabani [5] studies the number of possible requests as the parameter. In particular, the adversary can only generate requests from a fixed set of m requests known to the algorithm in advance. On

[1] The existential lower bound means that there exist n-point metrics on which the lower bound holds.
[2] The universal lower bound means that it holds for every n-point metric space.

uniform metrics, they showed that the deterministic competitive ratio is $\Theta(m \log(en/m))$, which is significantly better than the general bound of $\Theta(n)$. Interestingly, the randomized competitive ratio on uniform metrics is still $\Theta(\log n)$ even when $m = 2$. On the other hand, for two-level hierarchically separated tree (HST) metrics, the deterministic competitive ratio is $\Theta(n)$ and the randomized competitive ratio is $\Theta(\log n)$, even when m is constant. In summary, restricting m only improves the deterministic competitive ratio for uniform metrics, and does not help for two-level HSTs.

Parametrized MSS. The more interesting case is parametrized MSS, where parametrization leads to substantial improvements of the competitive ratio from the general case. For MSS, restricting possible requests within a fixed set of size m improves both deterministic and randomized competitive ratios: for uniform metrics, the deterministic and randomized competitive ratios are both $\Theta(m)$ and thus significantly better when $n \gg m$. Another interesting aspect of this result is that deterministic algorithms are essentially as powerful as randomized algorithms, which refutes the folklore that the randomized competitive ratio is polylogarithmic in the deterministic competitive ratio for most problems that can be interpreted as metrical task systems. As for the case of two-level HSTs, there are still gaps between the currently known upper and lower bounds: the deterministic competitive ratio is between $\Omega(\binom{m}{\lfloor m/2 \rfloor}) = \Omega(2^m/\sqrt{m})$ and $O(m2^m)$ and the randomized competitive ratio is at least $\Omega(2^{\lfloor m/2 \rfloor})$.

Bubeck and Rabani [5] proposed the following open problems in the study of parametrized MSS, which involve not only the exact tight bound of the competitive ratio in certain scenarios, but also understanding how restricting the number of possible requests m can help improve the competitive ratio, e.g., can we devise an online algorithm whose competitive ratio is purely in terms of m rather than metric space size n, and can we do this for some special classes of metrics like HSTs or even general metrics?

▶ **Question 1.** *What is the deterministic competitive ratio on weighted star metrics?*

▶ Remark 1.1. It is easy to design an $O(2^m)$-competitive deterministic algorithm and an $O(m)$-competitive randomized algorithm by reducing the problem to MSS on a weighted star where there are only $2^m - 1$ different states and applying the state-of-the-art result for MTS on weighted stars. For the special case of uniform metrics, the deterministic and randomized competitive ratios are both known to be $\Theta(m)$ so it is natural to ask if the deterministic algorithm can be generalized to weighted stars.

▶ **Question 2.** *What is the tight bound of the worst-case competitive ratio on all HSTs of two and higher levels?*

▶ **Question 3.** *For what constant values of m can we get constant-competitive algorithms for general metrics or some special classes of metrics like HSTs?*

1.1 Our Contributions

In this paper, we make substantial progress towards understanding the competitiveness of parametrized MSS.

Weighted stars. Our first main result answers Question 1 affirmatively.

▶ **Theorem 1.2.** *There is a $2m$-competitive deterministic algorithm for parameterized MSS on weighted stars.*

The result brings more insight into the foundation of this problem and the power of randomization as well, because it matches the lower bound of $\Omega(m)$ for randomized algorithms since the uniform metric is a special class of the weighted star. Theorem 1.2 shows that deterministic algorithms are essentially as powerful as randomized algorithms in achieving the best possible competitive ratio of $O(m)$ on weighted stars, the same as the case for the parametrized metrical service system problem on uniform metrics.

2-level HSTs. Next, we address Question 2 by settling the randomized competitive ratio on 2-level HSTs. Let $c_{k,m}$ denote the best possible randomized competitive ratio of the MSS problem that can be achieved on all k-level HSTs with m request types:

▶ **Theorem 1.3.** $c_{2,m} = \Theta(\binom{m}{\lceil m/2 \rceil}) = \Theta(2^m/\sqrt{m})$.

This theorem contains two separate claims: $c_{2,m} = O(\binom{m}{\lceil m/2 \rceil})$ (Theorem 4.17) and $c_{2,m} = \Omega(\binom{m}{\lceil m/2 \rceil})$ (Theorem 4.7). The lower bound is existential, in the sense that there exists a 2-level HST and m request types on it such that no randomized algorithm can achieve a better worst-case competitive ratio on all possible request sequences. Theorem 1.3 improves upon previous upper and lower bounds. Previously, the lower bound of $\Omega(\binom{m}{\lceil m/2 \rceil})$ is only known to hold for deterministic algorithms, and we prove that it also holds for randomized algorithms using a new hard instance.

Higher level HSTs. Finally, we turn to Question 3. Previously, Bubeck and Rabani [5] obtained lower bounds on $c_{k,m}$ for $k \geq 3$ by iterating their lower bound construction for $k = 2$. As a consequence, their lower bound rules out $O(1)$-competitive algorithms for $m \geq 6$. By iterating our construction in the proof of Theorem 4.7 to higher level HSTs, we also obtain improved lower bounds on $c_{k,m}$ for $k \geq 3$. The formal theorem statement (Theorem 4.8) is somewhat technical so we defer it to Section 4.1.4 and instead state the following corollary.

▶ **Corollary 1.4.** *For $m \geq 4$, there is no algorithm that achieves constant competitive ratio on HSTs.*

Combined with Observation 1.6, we conclude that the only open case for Question 3 on HSTs is $m = 3$.

General metrics. We resolve Question 3 on general metrics by the following two complement results.

We first show that a constant competitive ratio cannot be achieved on general metrics when m is at least 3 by the following existential lower bound of $\Omega(\log n)$:

▶ **Theorem 1.5.** *For $m \geq 3$, there exist $3n$-point metrics on which every randomized algorithm has competitive ratio of $\Omega(\log n)$.*

We refer the reader to the full version to see the proofs of Theorem 1.5 as well as the following Observation 1.6.

On the other hand, a constant competitive ratio can be achieved on general metrics when $m = 2$.

▶ **Observation 1.6.** *When $m = 2$, there is a 6-competitive deterministic algorithm on general metrics.*

This result follows from the observation that when $m = 2$, we can assume that the input sequence is strictly alternating between two request types, because MSS can be regarded as chasing feasible sets and hence repeating the same request does not incur additional cost. Therefore, the only uncertainty of the input is its total length, and the optimal cost can be expressed as a non-decreasing function of the input length. This observation allows us to apply the Guess-and-Double Method to design a constant-competitive algorithm for general metrics when $m = 2$.

Therefore, the open problems for Question 3 now lie on some special classes of metrics, including HSTs that we study in this paper.

1.2 Our Techniques

Weighted stars. In MSS, we can think of a request as the set of states that the server must move to. Bubeck and Rabani [5] use an algorithm that works in phases similar to the classical marking algorithm to provide a $O(m)$-deterministic algorithm for parametrized MSS on uniform metrics: in each phase, it moves to an arbitrary state in the intersection of the requests in the phase; if the intersection is empty, the algorithm starts a new phase and moves to an arbitrary requested state. It is clear that any solution needs to move at least once during each phase, and they show that the algorithm moves at most m times during each phase to prove the upper bound. This method is heavily based on the fact that the metric space is uniform and hence we can simply compare the number of times the algorithm's server moves with that of the optimal server. It is unclear whether their method can be adapted to solve the problem on weighted stars.

We use a significantly different approach based on an interval covering formulation of the problem and apply the primal-dual method. Each interval corresponds to a consecutive subsequence of requests, and the cost to buy this interval is the minimum possible cost of moving to a state that can satisfy all request types in this subsequence. Our analysis exploits the fact that there are only m different request types to show that the algorithmic cost is at most m times the feasible dual solution that we maintain and thus obtain the desired competitive ratio. Our formulation is similar to that in [6] but we exploit the property of MSS requests that allows us to only consider the unique state with minimum cost to satisfy every subsequence, while for MTS we have to consider various costs of serving the same subsequence of requests in n different states, thus leading to the appearance of n in the competitive ratio instead of purely in terms of m. We hope that similar methods under the linear programming and primal-dual framework can be adapted to solve the more general parametrized MTS problem.

Lower bounds on HSTs. We use the same high-level approach used by Bubeck and Rabani [5] in which a construction for the uniform metric[3] is lifted to HSTs of higher levels. The main conceptual difference is in our construction for the uniform metric. In Bubeck and Rabani's construction, there are $2^{m/2}$ points, each labeled by a node of the Boolean hypercube $\{0,1\}^{m/2}$ (we assume m is even here for simplicity). Each of the m requests correspond to a an axis-aligned halfspace of the Boolean hypercube. Thus, if we construct a sequence of $m/2$ requests by picking a random halfspace for each dimension with equal probability, then for each request, the algorithm has to move with probability $1/2$. On the other hand, the adversary can move to the point in the intersection of the halfspaces. This construction can be lifted to HSTs of higher levels by partitioning the set of all possible labels of the lower level case into pairs of antipodes and treating them as complementary requests.

[3] A 1-level HST is equivalent to a uniform metric.

Our key idea is to use a construction for the uniform metric that has more points. In particular, we use $\binom{m}{m/2}$ points, each corresponding to an $\frac{m}{2}$-sized subset of $[m]$. As a result, when we apply the above lifting process, for every fixed level k of the HST, there are more subtrees in our construction when m is sufficiently large. We need to be more careful in our analysis as it is no longer true that each request causes the algorithm to move with constant probability. Instead, we prove that its average is still lower bounded by a positive constant.

2 Preliminaries

Given a metric space $\mathcal{M} = (X, d)$ where X is the set of points and $d : X \times X \to \mathbb{R}_{\geq 0}$ is the distance function, a parametrized metrical service system instance consists of an initial location $s_0 \in X$ of the server and a sequence of requests $\{\rho_t\}_{t\geq 1}$, where each ρ_t is from a fixed collection $\{K_1, \ldots, K_m\}$ of m non-empty subsets of X. Upon the arrival of a request $\rho_t = K_i$ for some $1 \leq i \leq m$ at every step $t \geq 1$, one must move the server to some state $s_t \in K_i$ to maintain a feasible solution if its previous state s_{t-1} does not satisfy this, and the decision must be made without knowing future requests after ρ_t. As a result, MSS can be regarded as the online set chasing problem, illustrating that we always need to move the server to a state in the current requested set, and the parametrized setting restricts that there are only m different request types. The cost of a solution $\{s_t\}_{t\geq 1}$is defined to be

$$\sum_{t\geq 1} d(s_{t-1}, s_t),$$

where $d(s_{t-1}, s_t)$ denotes the transition cost to move from s_{t-1} to s_t upon the arrival of ρ_t. This can be interpreted as the total length of the trajectory of the server.

A randomized algorithm $\mathcal{A}$ has competitive ratio c if there exists some constant β that is independent of the input sequence (it might depend on the metric space $\mathcal{M}$) such that for every input sequence $\boldsymbol{\rho}$:

$$\mathbb{E}[\text{cost}_{\mathcal{A}}(\boldsymbol{\rho})] \leq c \cdot \text{cost}^*(\boldsymbol{\rho}) + \beta,$$

where $\mathbb{E}[\text{cost}_{\mathcal{A}}(\boldsymbol{\rho})]$ is the expected cost of $\mathcal{A}$ on input $\boldsymbol{\rho}$ and $\text{cost}^*(\boldsymbol{\rho})$ is the minimum cost among all feasible solutions of $\boldsymbol{\rho}$. If $\beta = 0$, then we say that c is a strict competitive ratio of $\mathcal{A}$. When we talk about the competitive ratio of a problem class (e.g. parametrized MSS with m requests on a class of metrics), we are referring to the best possible competitive ratio that can be achieved by some algorithm $\mathcal{A}$ on all possible inputs from this problem class.

3 Weighted Stars

On every weighted star metric, there is one root node to which every leaf node (state) is connected with a single edge of non-negative weight. Let d_s denote the weight of the edge between state s and the root, then the distance between two different states s_1 and s_2 is $d(s_1, s_2) = d_{s_1} + d_{s_2}$, where d_{s_1} can be charged as the cost of leaving state s_1 and d_{s_2} can be charged as that of entering state s_2 if the server moves from s_1 to s_2.

A common practice on weighted stars is that instead of charging the cost of both leaving s_1 and entering s_2, we only need to consider the cost of entering the new state at every step, i.e. we can regard the cost of moving from s_1 to a different state s_2 as only d_{s_2}, for both the algorithmic cost and the optimal solution. Over the entire process, the number of times that the server leaves any state is the same as the number of times that the server enters the same state, except for the initial and the last state. Now we formally show that this setting will only lose the competitive ratio of the original problem setting by a factor of at most 2.

▶ **Lemma 3.1.** *Every strictly c-competitive algorithm for the problem setting under the new transition costs has strict competitive ratio at most $2c$ in the original setting.*

Proof. Suppose that the initial state of the server is s_0, the algorithm moves to states $s_1, \ldots, s_n$ by the chronological order during its execution, and some fixed solution moves to states $s'_1, \ldots, s'_m$ one by one. By the fact that the algorithm achieves strict competitive ratio c under the new setting, it holds that $\sum_{j=1}^{n} d_{s_j} \leq c \sum_{j=1}^{m} d_{s'_j}$. Then we have $d_{s_0} + 2\sum_{j=1}^{n-1} d_{s_j} + d_{s_n} \leq d_{s_0} + 2\sum_{j=1}^{n} d_{s_j} \leq d_{s_0} + 2c\sum_{j=1}^{m} d_{s'_j} \leq 2c(d_{s_0} + 2\sum_{j=1}^{m-1} d_{s'_j} + d_{s'_m})$, which suggests that the same algorithm achieves strict competitive ratio $2c$ in the original setting by the arbitrariness of the solution that we are comparing the algorithmic cost with. ◀

State labels. In this setting, we do not need to consider the initial state. Moreover, each state can be labeled with a subset of $[m] = \{1, 2, \ldots, m\}$, which implies the set of request types that are feasible at this position. To formalize, given the label $S \subseteq [m]$ of some state s and $\{K_1, \ldots, K_m\}$ of the parametrized MSS setting of m possible request types as subsets of the metric space, $S = \{i \mid s \in K_i\}$. We may use the notations of a state s and its label S interchangeably, as well as a set of natural numbers and the set of request types K_i's whose indices are in this set.

Assumptions. It suffices to consider $2^m - 1$ states that correspond to $2^m - 1$ different non-empty subsets of $[m]$, since if there are at least two states with exactly the same label, then any reasonable algorithm should only visit the one whose edge to the node has the minimum weight. For every non-empty $S \subseteq [m] = \{1, 2, \ldots, m\}$, we use d_S to denote the minimum edge weight between the root and any node whose label is exactly equal to S. If no such node exists, just set $d_S = \infty$.

We now make another assumption, which is without loss of generality, yet important for the validity of the algorithm that we are going to propose.

▶ **Assumption 3.2.** *$d_S \leq d_{S'}$ for every $S \subseteq S'$.*

Otherwise, if there exist two different non-empty subsets S and S' of $[m]$ such that $S \subseteq S'$ yet $d_S > d_{S'}$, then every visit to state S can be replaced by a visit to state S', maintaining the feasibility because $S' \supseteq S$ while only reducing the cost. In this case, we can set d_S to $d_{S'}$ instead, which does not increase the optimal cost since we are only reducing edge lengths, while the algorithmic solution can replace each visit to S with one visit to S' in the original setting, which means that every algorithm can be transformed into one whose cost in the original setting is no more than that under this assumption.

We make one final assumption. Against any deterministic algorithm, one can assume that the adversary always forces the algorithm to move to a different state at every step as long as there is some K_i such that the current state of the algorithm is not in K_i, and the input ends when the current state is in the intersection of all K_i's for $1 \leq i \leq m$, if such a scenario exists. A straightforward consequence of this assumption on our problem setting is that any two consecutive requests from the input sequence are of different types, but we will eventually use a stronger argument that also exploits the algorithmic decisions seen by the adversary in the deterministic model to bound the desired competitive ratio.

Integer programming formulation. Under the input sequence $\{\rho_t\}_{t \geq 1}$, we use $S_{i,j}$ to denote the set of request types that appear in the subsequence $\rho_i, \ldots, \rho_j$, i.e. $S_{i,j} = \{\ell \mid K_\ell = \rho_t \text{ for some } i \leq t \leq j\}$, which also corresponds to a unique node of the weighted star in our

notation. Then we can write down an integer programming formulation of the problem whose minimum objective function value is a lower bound of the optimal solution of the original problem. We use T to express the number of steps of the whole input sequence.

$$\begin{aligned} \text{minimize} \quad & \sum_{1\le i\le j\le T} d_{S_{i,j}} x_{i,j} \\ \text{subject to} \quad & \sum_{i\le t\le j} x_{i,j} \ge 1, \quad \forall 1 \le t \le T \\ & x_{i,j} \in \{0,1\}, \quad \forall 1 \le i \le j \le T \end{aligned} \tag{1}$$

▶ **Lemma 3.3.** *The minimum value of (1) is no more than the optimal value of the parametrized MSS problem on the weighted star in our setting.*

Proof. Given any solution of the parametrized MSS problem on the weighted star, for every state that the server stays at from step i to j, we can just assume that it stays at state $S_{i,j}$ due to Assumption 3.2, otherwise the cost can only be higher. We set $x_{i,j} = 1$ for every such state that the server visits, then this is a feasible solution of (1), and its value is the same as the given solution of the parametrized MSS on the weighted star instance. ◀

The next step is to write down the dual program of the linear relaxation of (1), whose maximum value is no more than the minimum value of its primal linear program and thus the minimum value of the original program (1) as well:

$$\begin{aligned} \text{maximize} \quad & \sum_{1\le t\le T} y_t \\ \text{subject to} \quad & \sum_{t:i\le t\le j} y_t \le d_{S_{i,j}}, \forall 1 \le i \le j \le T \\ & y_t \ge 0, \quad \forall 1 \le t \le T \end{aligned} \tag{2}$$

Primal-dual algorithm. We are now ready to state our primal-dual algorithm. From a high level, our algorithm maintains a feasible dual solution and at every step t: increases the dual variable y_t until some dual constraint becomes tight, and then decides which state to move to depending on the tight dual constraint.

Now we come to its formal description. Upon the arrival of request ρ_t, increase the value of the dual variable y_t from the initial value 0 until some dual constraint is tight, i.e.

$$\sum_{\ell=i}^{t} y_\ell = d_{S_{i,t}}$$

for some $i \le \ell \le t$; or equivalently, we can write the value as

$$y_t = \min_{1\le i\le t} \left\{ d_{S_{i,t}} - \sum_{\ell=i}^{t-1} y_\ell \right\}, \tag{3}$$

where the value of $\sum_{\ell=i}^{t-1} y_\ell$ is defined to be zero for $i = t$.

If the tight dual constraint is $\sum_{\ell=i}^{t} y_\ell = d_{S_{i,t}}$, then the algorithm moves the server to the state labeled with $S_{i,t}$ (the one with the minimum edge weight to the root based on our previous assumption), to process request ρ_t. Ties are broken in an arbitrary way.

Algorithm analysis. We first prove that the algorithm indeed maintains a feasible dual solution.

▶ **Lemma 3.4.** *The algorithm maintains a feasible dual solution.*

Proof. At every step t, a new dual variable y_t whose initial value is zero appears with a set of new dual constraints, and we need to show that these constraints are not violated when $y_t = 0$, so that it is feasible to increase the value of y_t from zero until some dual constraint becomes tight.

It suffices to prove that $\sum_{\ell=i}^{t-1} y_\ell \le d_{S_{i,t}}$ for every $i \le t$, given the values y_1 to y_{t-1} well-defined from previous steps. The case that $i = t$ is trivial, since $d_{S_{i,t}}$ is non-negative based on the problem definition. For every $1 \le i \le t-1$, it holds that $\sum_{\ell=i}^{t-1} y_\ell \le d_{S_{i,t-1}}$ given that we maintain a feasible solution for dual variables and constraints that have appeared until step $t-1$, and the result follows by Assumption 3.2 and the fact that $S_{i,t-1} \subseteq S_{i,t}$. ◀

Now we are ready to use values of y_t's that the algorithm computes under its execution to provide an upper bound of its cost.

▶ **Lemma 3.5.** *The cost of the algorithm is at most* $m \sum_{\ell=1}^{T} y_\ell$.

Proof. When the algorithm moves to state $S_{i,t}$ at step t, $S_{i,t}$ is the set of request types in some subsequence $\rho_i, \ldots, \rho_t$ and the values of dual variables assigned by the algorithm so far satisfy that $\sum_{\ell=i}^{t} y_\ell = d_{S_{i,t}}$, since the algorithm chooses state $S_{i,t}$ to move to according to the tight dual constraint at every step. As a result, the cost of moving to state $S_{i,t}$ at every step t can be charged to some y_ℓ's. Now we show that every y_ℓ is only charged to at no more than m different steps t and their corresponding states $S_{i,t}$'s, which means that we need to show there are at most m pairs of (i,t)'s such that $i \le \ell \le t$ and the algorithm moves to state $S_{i,t}$ at step t.

Suppose there are two different steps $t_1 < t_2$ such that the algorithm moves to state S_{i_1,t_1} at step t_1 and state S_{i_2,t_2} at step t_2, and $i_1 \le \ell \le t_1$ and $i_2 \le \ell \le t_2$. The subsequence $\rho_{t_1+1}, \ldots, \rho_{t_2}$ contains at least one request that is not in set S_{i_1,t_1} – namely, ρ_{t_1+1} – as otherwise the algorithm has no incentive to move to a different state. As a result, since $S_{t_1+1,t_2} \subseteq S_{\ell,t_2}$ and $S_{\ell,t_1} \subseteq S_{i_1,t_1}$, we get $S_{\ell,t_2} \setminus S_{\ell,t_1} \supseteq S_{t_1+1,t_2} \setminus S_{i_1,t_1} \neq \varnothing$, and hence $|S_{\ell,t_2}| \ge |S_{\ell,t_1}| + 1$ because $S_{\ell,t_1} \subseteq S_{\ell,t_2}$. Now if we have in total k such different steps $\ell \le t_1 < \ldots < t_k$, then it holds that $|S_{\ell,t_k}| \ge k$, which directly implies that $k \le m$. ◀

Lemmas 3.1 and 3.5 and the fact that the maximum value of the linear dual program (2) is no more than the minimum value of the original program (1) imply Theorem 1.2.

4 Randomized Competitive Ratio on HSTs

In this section, we consider hierarchically separated trees (HSTs), a highly symmetric rooted tree structure where the edge weights between a node and all its children are equal. The path between one internal node to all leaves in the subtree rooted at this node has the same number of edges, meaning that the level of a node as the number of edges from it to any of its descendant leaf is well-defined (the level of every leaf node is 0). Moreover, the edge weight between an internal node and all its children is a function of the node level. Since we regard all leaf nodes as the points in the metric space induced by a rooted tree, the distance between two points is a function of the level of their lowest common ancestor. We say that an HST is a k-level HST if its root is at level k.

The bulk of this section is devoted to proving our lower bounds in Section 4.1. We finish with the tight upper bound for 2-level HSTs in Section 4.2.

4.1 Iterative Lower Bounds

The key ingredient for our improved lower bounds is a new construction for uniform metrics (i.e. 1-level HSTs). Thus, we begin in Section 4.1.1 by giving the construction and analysis for uniform metrics. Then, we show how to lift our ideas to construct higher level HSTs in Section 4.1.2. In Section 4.1.3, we prove our improved lower bound $c_{2,m} = \Omega(\binom{m}{\lceil m/2 \rceil})$ on 2-level HSTs. Finally, we prove our lower bounds for higher level HSTs.

Iterative sequence. We first introduce the following iterative sequence which will be used to express the constructions of hard instances and the lower bounds:

$$A_{1,m} = \binom{m}{\lceil m/2 \rceil}, \qquad A_{k,m} = \binom{A_{k-1,m}}{\lceil A_{k-1,m}/2 \rceil} \quad \text{for } k \geq 2. \tag{4}$$

▶ **Proposition 4.1.** *For every fixed $m \geq 4$, we have $A_{1,m} > m$ and so the sequence $\{A_{k,m}\}_{k \geq 1}$ is unbounded.*

Labels and requests. In this section, the requests are defined by assigning a label $S \subseteq [m]$ to each leaf node. The request type K_i (as a feasible set) is then defined to be the leaf nodes whose labels contain i, i.e. request type K_i is served by moving to a node v whose label S contains i. We may use number i to refer to request type K_i when the context is clear, especially when we use the corresponding subset of $[m]$ to refer to a set of request types.

Randomized lower bounds. To prove a lower bound of α on the competitive ratio of randomized online algorithms, we use Yao's minimax principle [17], and give a distribution over input instances such that every deterministic algorithm has expected competitive ratio at least α.

In the following, we will only describe and analyze a randomized input sequence whose length is a function of k and m. This process can be repeated to generate arbitrarily many rounds and hence the request sequence can be arbitrarily long and have arbitrarily large cost.

4.1.1 Warm-up: Uniform Metrics

We begin with a warm-up on a 1-level HST (uniform metric) which we will lift to higher level HSTs.

Lower bound instance. The key property of our 1-level HST is that for every request sequence that is a permutation of $[m]$, there is a leaf that can serve the first $\lceil \frac{m}{2} \rceil$ requests and another leaf that can serve the remaining requests, and thus there is always a solution that moves at most twice. Define the 1-level HST structure $\mathcal{T}_{1,m}$ as follows: it has $\binom{m}{\lceil m/2 \rceil} = A_{1,m}$ leaves, each labeled with a $\lceil \frac{m}{2} \rceil$-size subset of $[m]$ and the distance between every two leaves is 1. The adversarial request sequence is a uniform random permutation of $[m]$.

Before we start our analysis, we prove the following proposition which will be useful in bounding the expected cost of the algorithm.

▶ **Proposition 4.2.** *There exists a universal constant $\kappa > 0$ such that for every $M \geq 2$, we have*

$$\sum_{t=1}^{\lfloor M/2 \rfloor} \frac{\lfloor M/2 \rfloor - t + 1}{M - t + 1} \geq \kappa \cdot M.$$

Proof. We have

$$\begin{aligned}\sum_{t=1}^{\lfloor M/2\rfloor}\frac{\lfloor M/2\rfloor-t+1}{M-t+1} &= \sum_{t=1}^{\lfloor M/2\rfloor}\frac{(M-t+1)-\lceil M/2\rceil}{M-t+1}\\ &= \lfloor M/2\rfloor - \lfloor M/2\rfloor\sum_{i=\lceil M/2\rceil+1}^{M}\frac{1}{i}\\ &= \lfloor M/2\rfloor - \lceil M/2\rceil(H_M - H_{\lceil M/2\rceil})\\ &= \lfloor M/2\rfloor - \lceil M/2\rceil(\ln 2 + o(1)) = \frac{1-\ln 2}{2}M(1+o(1)),\end{aligned}$$

where H_i is the ith harmonic number. ◀

▶ **Theorem 4.3.** *The expected competitive ratio of every deterministic algorithm against the uniform random permutation of m request types on $\mathcal{T}_{1,m}$ is $\Omega(m)$.*

Proof. First, we show that there is a solution with cost at most 2. The whole sequence $\rho_1,\ldots,\rho_m$ has each of the possible request types $K_1,\ldots,K_m$ exactly once. The solution processes requests $\rho_1,\ldots,\rho_{\lceil m/2\rceil}$ at the state labeled by the set of all indices of request types in $\rho_1,\ldots,\rho_{\lceil m/2\rceil}$, i.e. at $S=\{i \mid K_i \text{ is in subsequence } \rho_1,\ldots,\rho_{\lceil m/2\rceil}\}$, and then moves to the state labeled by the set of all indices of request types in the subsequence $\rho_{\lfloor m/2\rfloor+1},\ldots,\rho_m$. The total cost of this solution is at most 2, no matter which initial state the server is in.

We now lower bound the expected cost of every deterministic algorithm. Fix a deterministic algorithm. For each step t, let R_t be the set of request types that have not appeared prior to this step and S_{t-1} denote the label of the state of the algorithm right before the arrival of ρ_t. It holds that $|R_t| = m-t+1$ and $|S_{t-1}| = \lceil m/2\rceil$, and hence $|R_t \setminus S_{t-1}| \geq \lfloor m/2\rfloor - t + 1$. Let Y_t denote the event that the algorithm has to move at step t (i.e. because $\rho_t \notin S_{t-1}$). Since ρ_t is chosen u.a.r. from R_t, we have $\mathbb{E}[Y_t] \geq (\lfloor m/2\rfloor - t + 1)/(m-t+1)$ for every $1 \leq t \leq \lfloor m/2\rfloor$. For $\lfloor m/2\rfloor + 1 \leq t \leq m$, we use the trivial bound $\mathbb{E}[Y_t] \geq 0$. Thus, the total expected cost of the deterministic algorithm is

$$\sum_{t=1}^{m}\mathbb{E}[Y_t] \geq \sum_{t=1}^{\lfloor m/2\rfloor}\frac{\lfloor m/2\rfloor-t+1}{m-t+1} = \Omega(m),$$

by Proposition 4.2. ◀

4.1.2 Lifting to Higher Levels

For every k and m, we construct an HST $\mathcal{T}_{k,m}$ to lower bound $c_{k,m}$.

Key property of $\mathcal{T}_{k,m}$. The key property of $\mathcal{T}_{1,m}$ and the distribution of request sequences is that for every request sequence in the support of the distribution, there is a leaf that can serve the first half of the requests, and another leaf that can serve the second half. Thus, there is a solution that only moves twice. We generalize this to $\mathcal{T}_{k,m}$ as follows. The distribution of request sequences and $\mathcal{T}_{k,m}$ have the property that for every sequence in the support of the distribution, there are two "safe" $(k-1)$-level subtrees such that there is a low-cost solution that can serve the first half of the request sequence in one of them and the second half in the other. In other words, the low-cost solution only moves twice between $(k-1)$-level subtrees.

Higher-level labels. In our construction of $\mathcal{T}_{k,m}$, we will assign labels to all nodes except the root. The labels for leaves come from the set $\mathcal{P}_0 := \binom{[m]}{\lceil m/2 \rceil}$[4]. The labels for ℓ-level nodes are collections of those for $(\ell-1)$-level nodes as follows: Let $\mathcal{P}_\ell$ be the set of ℓ-level labels, then $\mathcal{P}_\ell := \binom{\mathcal{P}_{\ell-1}}{\lceil |\mathcal{P}_{\ell-1}|/2 \rceil}$ for every $1 \le \ell \le k-1$. It is easy to see that $|\mathcal{P}_\ell| = A_{\ell+1,m}$. We also define the label of a subtree to be the label of its root node.

Construction of $\mathcal{T}_{k,m}$. For $k \ge 2$, the HST $\mathcal{T}_{k,m}$ is constructed iteratively in a top-down fashion as follows. For every label $P \in \mathcal{P}_{k-1}$, the root of the HST has a child labeled P. For every level $k-1 \ge \ell \ge 1$, every level-ℓ node v has $|P(v)|$ children where $P(v)$ is its label, each given a unique label $p \in P(v)$.[5] We remark that there is exactly one level-$(k-1)$ node per label in $\mathcal{P}_{k-1}$ but for $\ell < k-1$, there are several level-ℓ nodes from different $(\ell+1)$-level subtrees that share the same label. For every $1 \le \ell \le k$, the distance between every two leaves whose lowest common ancestor is on level ℓ is $C_{\ell-1}$[6] ($C_0 = 1$), where C_ℓ's depend on k and m and will be defined later in Section 4.1.4.

▶ **Observation 4.4.** *In $\mathcal{T}_{k,m}$, every ℓ-level node has exactly $\lceil A_{\ell,m}/2 \rceil$ level-$(\ell-1)$ children with distinct labels, for every $1 \le \ell \le k-1$.*

4.1.3 Lower Bound Sequence for $\mathcal{T}_{2,m}$

We now prove our lower bound of $c_{2,m}$. Recall that C_1 is the distance between two leaves in different 1-level subtrees, and $\mathcal{P}_0$ is the set of all $\lceil m/2 \rceil$-size subsets of $[m]$. For each label $S \in \mathcal{P}_0$, define $\boldsymbol{\rho}(S)$ to be an arbitrary permutation of S.

Now we describe the adversarial randomized input sequence. The adversary first samples a uniformly random permutation $\{S_t\}_{t=1}^{|\mathcal{P}_0|}$ of $\mathcal{P}_0$ and then proceeds in $|\mathcal{P}_0| = A_{1,m}$ phases as follows: in phase t, it repeats the sequence $\boldsymbol{\rho}(S_t)$ for C_1 times. Thus, the entire input is a concatenation of all those phases:

$$\underbrace{\boldsymbol{\rho}(S_1), \ldots, \boldsymbol{\rho}(S_1)}_{C_1 \text{ times}}, \underbrace{\boldsymbol{\rho}(S_2), \ldots, \boldsymbol{\rho}(S_2)}_{C_1 \text{ times}}, \ldots, \underbrace{\boldsymbol{\rho}(S_{|\mathcal{P}_0|}), \ldots, \boldsymbol{\rho}(S_{|\mathcal{P}_0|})}_{C_1 \text{ times}}.$$

For every $1 \le t \le A_{1,m}$, we use T^1_{t-1} to denote the label of the 1-level subtree in which the state of the algorithm is located right before the beginning of the phase t.

▶ **Lemma 4.5.** *For every C_1, every deterministic algorithm has expected cost of at least $\Omega(A_{1,m}C_1)$.*

Proof. We first show that: the cost incurred during the whole phase t is at least C_1 if T^1_{t-1} does not contain S_t. If $S_t \notin T^1_{t-1}$, either the algorithm moves to another 1-level subtree that contains S_t as a leaf node, or the algorithm needs to move at least once during every loop of $\boldsymbol{\rho}(S_t)$ since there is no state in T^1_{t-1} that can satisfy all request types in S_t. In either way, the cost is at least C_1.

Now we consider the probability of the event that T^1_{t-1} does not contain S_t. Similar to the proof of Theorem 4.3, at every phase $1 \le t \le \lfloor A_{1,m}/2 \rfloor$, the probability that T^1_{t-1} does not contain a node labeled S_t is at least $(\lfloor A_{1,m}/2 \rfloor - t + 1)/(A_{1,m} - t + 1)$ because T^1_{t-1} contains exactly $\lceil A_{1,m}/2 \rceil$ different leaf labels. So, Proposition 4.2 implies that the total expected cost incurred during the first $\lfloor A_{1,m}/2 \rfloor$ phases is $\Omega(A_{1,m}C_1)$. ◀

---

[4] The notation $\binom{S}{k}$ means the collection of all k-size subsets of S.
[5] Recall that $P(v)$, a label of a level-ℓ node, is a collection of level-$(\ell-1)$ labels.
[6] We always assume that $C_\ell/C_{\ell-1}$ is an integer for every $\ell \ge 1$.

▶ **Lemma 4.6.** *For every C_1, the optimal offline cost is at most $2C_1 + A_{1,m}$ no matter where the initial state is.*

Proof. Consider the following solution. It moves to the subtree labeled $\{S_1, \ldots, S_{\lceil A_{1,m}/2 \rceil}\}$ and for phases $1 \le t \le \lfloor A_{1,m}/2 \rfloor$, it serves the repetitions of the subsequence $\boldsymbol{\rho}(S_t)$ at the node labeled S_t in the subtree. Then, it moves to the subtree labeled $\{S_{\lfloor A_{1,m}/2 \rfloor + 1}, \ldots, S_{A_{1,m}}\}$ and for $\lfloor A_{1,m}/2 \rfloor < t \le A_{1,m}$, serves the repetitions of the subsequence $\boldsymbol{\rho}(S_t)$ at the node labeled S_t in the subtree. It incurs a cost of at most $2C_1$ for movement between different 1-level subtrees and at most $A_{1,m}$ for movement within each of the two 1-level subtrees. Thus, the total cost is at most $2C_1 + A_{1,m}$. ◀

Since the two lemmata that we have just proven hold for every value C_1, we can set it to be some value such that $C_1 = \omega(A_{1,m})$ with both values regarded as functions of m, and the desired lower bound immediately follows.

▶ **Theorem 4.7.** *Every randomized algorithm has a worst case competitive ratio $\Omega(A_{1,m})$ on some instances of 2-level HSTs.*

4.1.4 General Iterative Lower Bounds on Higher Levels

We now prove the lower bound on $c_{k,m}$ for $k \ge 3$ expressed with the iterative sequence $\{A_{k,m}\}_{k \ge 1}$ defined in Equation (4) at the beginning of Section 4.1.

▶ **Theorem 4.8.** *$c_{k,m} = \Omega(A_{k-1,m})$ for every $k \ge 3$.*

Corollary 1.4 now follows from Theorem 4.8 and Proposition 4.1.

Sequences of labels. We have provided the construction of $\mathcal{T}_{k,m}$ for every k, and now we need to describe the adversary's strategy to generate a randomized input sequence on which no deterministic algorithm achieves competitive ratio better than $c_{k,m}$. Recall that every sequence in the support of the randomized input on $\mathcal{T}_{2,m}$ is a concatenation of sequences $\boldsymbol{\rho}(S)$ for labels $S \in \mathcal{P}_0$. Now we want to generalize this concept, the sequence of a label, to every level-ℓ label (i.e. every element of $\mathcal{P}_l$).

We define inductively on ℓ with $\ell = 0$ as the base case. For every $\ell \ge 1$ and every label $T^\ell \in \mathcal{P}_\ell$, the sequence $\boldsymbol{\rho}(T^\ell)$ is constructed in $|T^\ell|$ phases as follows. Let $T_1^{\ell-1}, \ldots, T_{|T^\ell|}^{\ell-1}$ be an arbitrary permutation of T^ℓ. During each phase $1 \le t \le |T^\ell|$, the sequence $\boldsymbol{\rho}(T^{\ell-1})$ is repeated consecutively for $\frac{C_\ell}{C_{\ell-1}}$ (we can choose values of C_ℓ's to make this amount always an integer at the end) times, and the entire sequence $\boldsymbol{\rho}(T^\ell)$ is a concatenation of those phases:

$$\underbrace{\boldsymbol{\rho}(T_1^{\ell-1}), \ldots, \boldsymbol{\rho}(T_1^{\ell-1})}_{\frac{C_\ell}{C_{\ell-1}} \text{ times}}, \underbrace{\boldsymbol{\rho}(T_2^{\ell-1}), \ldots, \boldsymbol{\rho}(T_2^{\ell-1})}_{\frac{C_\ell}{C_{\ell-1}} \text{ times}}, \ldots, \underbrace{\boldsymbol{\rho}(T_{|T^\ell|}^{\ell-1}), \ldots, \boldsymbol{\rho}(T_{|T^\ell|}^{\ell-1})}_{\frac{C_\ell}{C_{\ell-1}} \text{ times}}.$$

▶ **Lemma 4.9.** *Let $1 \le \ell \le k-2$ and $P \in \mathcal{P}_\ell$. Consider a solution for $\boldsymbol{\rho}(P)$ whose initial state is not in an ℓ-level subtree with label P. Then, the solution incurs a cost of at least C_ℓ on $\boldsymbol{\rho}(P)$.*

Proof. We prove by induction on ℓ. The base case is when $\ell = 1$. Fix some $P \in \mathcal{P}_1$. The sequence $\boldsymbol{\rho}(P)$ consists of C_1 consecutive repetitions of the sequence $\boldsymbol{\rho}(p)$ for every child $p \in P$. If the initial state is in a 1-level subtree labeled $T^1 \ne P$, then there exists some $p \in P$ such that $p \notin T^1$ and thus there is no leaf in the subtree labeled p. As a result, the algorithm either visits a different 1-level subtree, or incurs cost 1 during every repetition of $\boldsymbol{\rho}(p)$, thus costing at least C_1.

Suppose the claim holds for every sequence $\boldsymbol{\rho}(p)$ such that $p \in \mathcal{P}_\ell$ $(1 \le \ell \le k-3)$. Now consider a sequence $\boldsymbol{\rho}(P)$ for some $P \in \mathcal{P}_{\ell+1}$. For every $p \in P$, it contains $\frac{C_{\ell+1}}{C_\ell}$ consecutive repetitions of $\boldsymbol{\rho}(p)$. If the initial state is in an $(\ell+1)$-level subtree labeled $T^{\ell+1} \neq P$, then there exists $p \in P \setminus T^{\ell+1}$ because $|T^{\ell+1}| = |P|$. If the algorithm moves to a different $(\ell+1)$-level subtree, it incurs a cost of $C_{\ell+1}$. Now suppose the algorithm does not move to a different $(\ell+1)$-level subtree. Since $p \notin T_{\ell+1}$, the current subtree does not have an ℓ-level subtree labeled p. Thus at the start of each of the $C_{\ell+1}/C_\ell$ repetitions of $\boldsymbol{\rho}(p)$, by the inductive hypothesis, it incurs a cost of at least C_ℓ. The total cost is thus at least $C_{\ell+1}$. ◀

We now define another iterative sequence $\{B_\ell\}_{1 \le \ell \le k-1}$ and show that it can be used to express the upper bound of the optimal solution of the request sequences:

$$B_1 = \left\lceil \frac{A_{1,m}}{2} \right\rceil, \qquad B_\ell = \left\lceil \frac{A_{\ell,m}}{2} \right\rceil \left[(B_{\ell-1} + C_{\ell-2}) \cdot \frac{C_\ell}{C_{\ell-1}} + C_{\ell-1} \right] \quad \text{for } 2 \le \ell \le k-1.$$

▶ **Lemma 4.10.** *For every label T^ℓ in $\mathcal{P}_\ell$ and subtree labeled T^ℓ in $\mathcal{T}_{k,m}$, there exists a trajectory contained within the subtree that incurs cost of no more than B_ℓ to serve the sequence $\boldsymbol{\rho}(T^\ell)$, for every $1 \le \ell \le k-1$.*

Proof. We prove by induction on ℓ. Consider the base case $\ell = 1$. Fix some $T^1 \in \mathcal{P}_1$. Order the elements $S_1, \ldots, S_{\lceil A_{1,m}/2 \rceil}$ of T in the same order as the phases in $\boldsymbol{\rho}(T^1)$. Consider the trajectory that first moves to the leaf node labeled S_1 in the 1-level subtree labeled T^1 and then moves to leaves $S_2, \ldots, S_{\lceil A_{1,m}/2 \rceil}$ one by one with cost 1 each time since they are all in the same 1-level subtree. The total cost is at most $\left\lceil \frac{A_{1,m}}{2} \right\rceil = B_1$.

Now we consider $\boldsymbol{\rho}(T^\ell)$ for $2 \le \ell \le k-1$, given that the property for $\ell-1$ has been proven. Suppose that $\boldsymbol{\rho}(T^\ell)$ is the concatenation of $\boldsymbol{\rho}(T_1^{\ell-1}), \ldots, \boldsymbol{\rho}(T^{\ell-1}_{\lceil A_{\ell,m}/2 \rceil})$ by the chronological order where each $\boldsymbol{\rho}(T_j^{\ell-1})$ is repeated for $\frac{C_\ell}{C_{\ell-1}}$ times before proceeding to the next one. Consider the trajectory that for each phase $1 \le j \le \lceil A_{\ell,m}/2 \rceil$, consists of $\frac{C_\ell}{C_{\ell-1}}$ copies of the trajectory obtained by applying the inductive hypothesis to $\boldsymbol{\rho}(T_j^{\ell-1})$. In each phase j, for each copy of $\boldsymbol{\rho}(T_j^{\ell-1})$, the trajectory incurs a cost of at most $C_{\ell-2}$ to move to its initial state within the subtree with label $T_j^{\ell-1}$ and then a cost of $B_{\ell-1}$ to serve $\boldsymbol{\rho}(T_j^{\ell-1})$ within the same subtree. Thus, for each phase j, the trajectory's cost is at most $\frac{C_\ell}{C_{\ell-1}}(B_{\ell-1} + C_{\ell-2})$ plus at most $C_{\ell-1}$ to move to an $(\ell-1)$-level subtree labeled $T_j^{\ell-1}$ at the start of the phase. As there are $\lceil A_{\ell,m}/2 \rceil$ phases, the total cost of the trajectory is bounded by B_ℓ, as desired. ◀

Adversary's strategy and analysis. We are finally ready to describe the adversary's strategy on $\mathcal{T}_{k,m}$. It first samples a uniformly random permutation $\{T_t^{k-2}\}_{t=1}^{A_{k-1,m}}$ of $\mathcal{P}_{k-2}$ (note that $|\mathcal{P}_{k-2}| = A_{k-1,m}$), and then proceeds in $A_{k-1,m}$ phases: in each phase t, $\boldsymbol{\rho}(T_t^{k-2})$ is repeated consecutively for $\frac{C_{k-1}}{C_{k-2}}$ times, and the whole input is the concatenation of all $A_{k-1,m}$ phases.

▶ **Lemma 4.11.** *The optimal cost is no more than $2(C_{k-1} + B_{k-1})$ regardless of the initial state.*

Proof. The whole sequence can be regarded as the concatenation of the sequences $\boldsymbol{\rho}(T^{k-1})$'s of two $(k-1)$-level subtrees:

$$\{T_1^{k-2}, \ldots, T^{k-2}_{\lceil A_{k-1,m}/2 \rceil}\} \text{ and } \{T^{k-2}_{\lfloor A_{k-1,m}/2 \rfloor + 1}, \ldots, T^{k-2}_{A_{k-1,m}}\}.$$

The repetitions of $\boldsymbol{\rho}(T^{k-2}_{\lfloor A_{k-1,m}/2 \rfloor + 1})$ appear in both of the $(k-1)$-level subtrees' sequence when m is odd, but it does not affect our conclusion since we are proving an upper bound. As a result, the optimal solution just pays the cost at most B_{k-1} in each T^{k-1} (Lemma 4.10) plus the cost at most $2C_{k-1}$ to visit two $(k-1)$-level subtrees. ◀

▶ **Lemma 4.12.** *The expected cost of every deterministic algorithm to serve this randomized input on $\mathcal{T}_{k,m}$ is $\Omega(A_{k-1,m}C_{k-1})$.*

Proof. The proof is similar those of Theorem 4.3 and Lemma 4.5. Fix a deterministic algorithm. During every phase $1 \le t \le \lfloor A_{k-1,m}/2 \rfloor$ when some $\boldsymbol{\rho}(T_t^{k-2})$ is repeated for $\frac{C_{k-1}}{C_{k-2}}$ times, the probability that $\boldsymbol{\rho}(T_t^{k-2})$ is not in the current $(k-1)$-level subtree containing the algorithm's server has probability is at least $\frac{\lfloor A_{k-1,m}/2 \rfloor - t+1}{A_{k-1,m}-t+1}$. When this happens, the algorithm only has two choices: either move to a new $(k-1)$-level subtree with cost C_{k-1}, or incur cost of at least C_{k-2} during each repetition of $\boldsymbol{\rho}(T_t^{k-2})$ according to Lemma 4.9 since the initial state cannot be in a $(k-2)$-level subtree with label T_t^{k-2}. ◀

It remains to show that there exist possible values of C_ℓ's and B_ℓ's for $1 \le \ell \le k-1$ such that $C_{k-1} = \omega(B_{k-1})$ as m tends to infinity[7]. This would show that the randomized competitive ratio on $\mathcal{T}_{k,m}$ is at least $\Omega(\frac{A_{k-1,m}C_{k-1}}{C_{k-1}+B_{k-1}}) = \Omega(A_{k-1,m})$ as a function of m.

Let $C_1 = \omega(\prod_{j=1}^{k-1} A_{j,m})$ and $C_\ell = \omega((\prod_{j=\ell}^{k-1} A_{j,m})C_{\ell-1})$ for $2 \le \ell \le k-1$. We now prove that such a sequence satisfies the desired property.

▶ **Lemma 4.13.** $C_{k-1} = \omega(B_{k-1})$.

Proof. We prove that $C_\ell = \omega((\prod_{j=\ell+1}^{k-1} A_{j,m})B_\ell)$ as m tends to infinity for every $1 \le \ell \le k-1$ inductively, where $\prod_{j=k}^{k-1} A_{j,m}$ is defined to be 1.

Clearly the result holds for $\ell = 1$ because $B_1 = \Theta(A_{1,m})$. With the inductive hypothesis that $C_\ell = \omega((\prod_{j=\ell+1}^{k-1} A_{j,m})B_\ell)$ for some $1 \le \ell \le k-2$, then in order to prove that $C_{\ell+1} = \omega((\prod_{j=\ell+2}^{k-1} A_{j,m})B_{\ell+1})$, according to the definition that $B_{\ell+1} = \left\lceil \frac{A_{\ell+1,m}}{2} \right\rceil [(B_\ell + C_{\ell-1}) \cdot \frac{C_{\ell+1}}{C_\ell} + C_\ell]$, it suffices to show that the following two statements both hold:

$$C_{\ell+1} = \omega(A_{\ell+1,m} \cdot (\prod_{j=\ell+2}^{k-1} A_{j,m})C_\ell) = \omega(\prod_{j=\ell+1}^{k-1} A_{j,m}C_\ell), \tag{5}$$

which is exactly our definition of $C_{\ell+1}$, and

$$C_\ell = \omega((\prod_{j=\ell+1}^{k-1} A_{j,m})(B_\ell + C_{\ell-1})), \tag{6}$$

while (6) can be further reduced to that the following two formulae $C_\ell = \omega((\prod_{j=\ell+1}^{k-1} A_{j,m})B_\ell)$ and $C_\ell = \omega((\prod_{j=\ell+1}^{k-1} A_{j,m})C_{\ell-1})$ both hold, where the first one holds due to the inductive hypothesis and the second holds due to the definition of C_ℓ. ◀

This completes the proof of Theorem 4.8.

▶ Remark 4.14. Our iterative sequence of lower bounds on k-level HSTs marks an improvement from the one provided in Section 4.2 of [5] in the sense that: given every fixed $k \ge 2$, the lower bound provided by our sequence is asymptotically tighter than theirs when the parameter m tends to infinity. To formalize, consider their iterative sequence $\{\hat{c}_{k,m}\}_{k \ge 1}$ defined as the following: $\hat{c}_{1,m} = \lfloor m/2 \rfloor$ and $\hat{c}_{k,m} = 2^{\hat{c}_{k-1,m}-1}$. We can use inductive arguments on k to show that, $A_{k,m} = \omega(\hat{c}_{k,m})$ for every $k \ge 2$: it is clear that the statement

[7] Here both C_{k-1} and B_{k-1} are functions of m. When we consider both m and k as constants, the lower bound can be achieved by setting the ratio $C_\ell/C_{\ell-1}$ sufficiently large for every $1 \le \ell \le k-1$ and the proof idea is similar.

holds for $k = 2$, since $A_{2,m} = \binom{m}{\lceil m/2 \rceil} = \Theta(2^m/\sqrt{m})$ and $\hat{c}_{2,m} = \Theta(2^{m/2})$; suppose that $\lim_{m\to\infty} A_{k-1,m}/\hat{c}_{k-1,m} = \infty$, then we show that $\lim_{m\to\infty} \log(A_{k,m}/\hat{c}_{k,m}) = \infty$ and hence $\lim_{m\to\infty} A_{k,m}/\hat{c}_{k,m} = \infty$. To see this, $\log(A_{k,m}/\hat{c}_{k,m}) = \log A_{k,m} - \log \hat{c}_{k,m} = A_{k-1,m} - \log(A_{k-1,m})/2 - \hat{c}_{k-1,m} + O(1)$. Due to the inductive hypothesis, $\hat{c}_{k-1,m} = o(A_{k-1,m})$. Therefore, $A_{k-1,m} - \log(A_{k-1,m})/2 - \hat{c}_{k-1,m} \sim A_{k-1,m}$ and $\lim_{m\to\infty} \log(A_{k,m}/\hat{c}_{k,m}) = \infty$.

4.2 Tight Upper Bound for 2-Level HSTs

The idea to show a general upper bound for 2-level HSTs is as follows: even if there is no restriction on the size of the whole tree itself, some leaves and subtrees are redundant in the sense that any reasonable solution should not move to them, or does not need to move to them while maintaining feasibility and not increasing the cost. As a result, we can trim off those redundant nodes and hence reduce the graph size, just as how we show that we only need to consider $2^m - 1$ states for the weighted star case. This idea has also been applied by Bubeck and Rabani to give iterative upper bounds for 2 and higher levels of HSTs (see Section 4.2 Deeper Trees in [5]), and we present a refined argument on 2-level HSTs that closes the gap between upper and lower bounds. We show that the number of leaves remained after such preprocessing is bounded in terms of m and apply the optimal MTS algorithm for HSTs from [4].

Since the initial state can be arbitrarily given, in this part we only consider those leaves different from the initial state, to evaluate whether some states can be totally replaced by others and every algorithm can still maintain a feasible solution with equal or less cost. This is without loss of generality up to a constant factor, since there is only one initial state and thus it is sufficient to give an upper bound of the amount of states that are not the initial state.

In this part, we express leaf nodes lexicographically by a tuple (S, T) where S is the label of the leaf node itself and T is the label of the 1-level subtree that it is in. The label of every node is defined in the same way as the previous subsection[8] . A preliminary observation is that, if two leaf nodes share exactly the same tuple, then they are isomorphic and it suffices to only keep one in the tree. The feasibility of removing redundant nodes is that, when m possible requests and their corresponding feasible sets in the whole metric are fixed, running an algorithm over a subset of states produces a feasible solution for the whole metric as well, therefore what we need to show is that the optimal cost does not increase while we trim off some states.

▶ **Proposition 4.15.** *If there are two leaves (S_1, T) and (S_2, T) in the same 1-level subtree such that $S_1 \subseteq S_2$, then any move to (S_1, T) can be replaced with one to (S_2, T) without increasing the cost.*

Proof. Any request that is feasible at (S_1, T) is also feasible at (S_2, T) since $S_1 \subseteq S_2$. Moreover, for any state different from (S_1, T) and (S_2, T), it has the same distance from those two nodes that are within the same 1-level subtree T according to the 2-level HST structure. ◀

As a result, we can assume that every 1-level subtree is an antichain over subsets of $[m]$, which contains at most $\binom{m}{\lceil m/2 \rceil} = A_{1,m}$ elements according to Sperner's theorem [16]. The next thing is to consider the maximum amount of such 1-level subtrees, which is equivalent to the amount of antichains over subsets of $[m]$. This amount is called Dedekind number $D(m)$ and its following asymptotic approximation property has been proven in [12]:

[8] Recall that the label of a leaf node is the set of request types that this state can satisfy, and the label of a 1-level subtree is the set of labels of leaf nodes contained in it.

▶ **Lemma 4.16** ([12]). $\log D(m) = O(A_{1,m})$.

▶ **Theorem 4.17.** $c_{2,m} = O(A_{1,m})$.

Proof. It suffices to consider at most $O(D(m)A_{1,m})$ leaf nodes (including the initial state and at most $A_{1,m}$ other leaves in the same 1-level subtree as they cannot be pruned by the above process) and apply the optimal online MTS algorithm on an HST that has competitive ratio $O(\log n)$, where n is the number of points in the metric. According to the asymptotic property of $D(m)$ as shown in Lemma 4.16, $\log(D(m)A_{1,m}) = O(A_{1,m} + \log A_{1,m}) = O(A_{1,m})$. ◀

5 Open Problems

Following the initial study of parametrized metrical task systems (MTS) and metrical service systems (MSS) by Bubeck and Rabani [5], our research answers several open problems posed by Bubeck and Rabani. We provide a deterministic $O(m)$-competitive algorithm on weighted stars, tighten the gap between the upper and lower bounds of randomized competitive ratios on HSTs of different levels, and show that it is possible to design a strictly constant-competitive algorithm on general metrics when $m = 2$. However, multiple problems remain open: as Question 3 has been resolved on general metrics by showing the contrast between $m = 2$ and $m = 3$ cases, the focus is now on special classes of metrics, e.g., is it possible to obtain a finite competitive ratio for chasing 3 sets on HSTs or line metrics? Another interesting open problem is whether the $O(\binom{m}{\lceil m/2 \rceil})$ upper bound of randomized algorithms on 2-level HSTs can also be achieved by a deterministic algorithm. The progress on this problem will not only close the gap between the upper and lower bounds known for deterministic algorithms for this particular class of metrics, but also deepen our fundamental understanding of the parametrized MSS problem, as it concerns whether deterministic algorithms are essentially as powerful as randomized algorithms on 2-level HSTs and other classes of metrics. Moreover, the technique that we use in Section 3 is a linear programming primal-dual method that exploits the properties of parametrized MSS with m request types, and we wonder whether this framework can be generalized to prove the results of the more general parametrized MTS on uniform metrics (the tight deterministic bound of $\Theta(m \log(en/m))$ was proven by Bubeck and Rabani) or weighted stars (no result yet).

References

1 John Augustine, Sandy Irani, and Chaitanya Swamy. Optimal power-down strategies. *SIAM J. Comput.*, 37(5):1499–1516, 2008. doi:10.1137/05063787X.

2 Allan Borodin, Nathan Linial, and Michael E. Saks. An optimal online algorithm for metrical task systems. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 373–382. ACM, 1987. doi:10.1145/28395.28435.

3 Sébastien Bubeck, Christian Coester, and Yuval Rabani. The randomized k-server conjecture is false! In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 581–594. ACM, 2023. doi:10.1145/3564246.3585132.

4 Sébastien Bubeck, Michael B. Cohen, James R. Lee, and Yin Tat Lee. Metrical task systems on trees via mirror descent and unfair gluing. *SIAM J. Comput.*, 50(3):909–923, 2021. doi:10.1137/19M1237879.

5 Sébastien Bubeck and Yuval Rabani. Parametrized metrical task systems. In Jaroslaw Byrka and Raghu Meka, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference*, volume 176 of *LIPIcs*, pages 54:1–54:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.54.

6 Niv Buchbinder and Joseph Naor. Online primal-dual algorithms for covering and packing. *Math. Oper. Res.*, 34(2):270–286, 2009. doi:10.1287/MOOR.1080.0363.

7 William R. Burley. Traversing layered graphs using the work function algorithm. *J. Algorithms*, 20(3):479–511, 1996. doi:10.1006/JAGM.1996.0024.

8 William R. Burley and Sandy Irani. On algorithm design for metrical task systems. *Algorithmica*, 18(4):461–485, 1997. doi:10.1007/PL00009166.

9 Marek Chrobak and Lawrence L. Larmore. The server problem and on-line games. In Lyle A. McGeoch and Daniel Dominic Sleator, editors, *On-Line Algorithms, Proceedings of a DIMACS Workshop, New Brunswick, New Jersey, USA, February 11-13, 1991*, volume 7 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 11–64. DIMACS/AMS, 1991. doi:10.1090/DIMACS/007/02.

10 Amos Fiat, Dean P. Foster, Howard J. Karloff, Yuval Rabani, Yiftach Ravid, and Sundar Vishwanathan. Competitive algorithms for layered graph traversal. In *32nd Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, October 1-4, 1991*, pages 288–297. IEEE Computer Society, 1991. doi:10.1109/SFCS.1991.185381.

11 Sandy Irani, Sandeep K. Shukla, and Rajesh K. Gupta. Online strategies for dynamic power management in systems with multiple power-saving states. *ACM Trans. Embed. Comput. Syst.*, 2(3):325–346, 2003. doi:10.1145/860176.860180.

12 Daniel Kleitman and George Markowsky. On dedekind's problem: the number of isotone boolean functions. ii. *Transactions of the American Mathematical Society*, 213:373–390, 1975.

13 Mark S. Manasse, Lyle A. McGeoch, and Daniel Dominic Sleator. Competitive algorithms for server problems. *J. Algorithms*, 11(2):208–230, 1990. doi:10.1016/0196-6774(90)90003-W.

14 H. Ramesh. On traversing layered graphs on-line. In Vijaya Ramachandran, editor, *Proceedings of the Fourth Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 25-27 January 1993, Austin, Texas, USA*, pages 412–421. ACM/SIAM, 1993. URL: http://dl.acm.org/citation.cfm?id=313559.313843.

15 Daniel Dominic Sleator and Robert Endre Tarjan. Amortized efficiency of list update and paging rules. *Commun. ACM*, 28(2):202–208, 1985. doi:10.1145/2786.2793.

16 Emanuel Sperner. Ein satz über untermengen einer endlichen menge. *Mathematische Zeitschrift*, 27(1):544–548, 1928.

17 Andrew Chi-Chih Yao. Probabilistic computations: Toward a unified measure of complexity (extended abstract). In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*, pages 222–227. IEEE Computer Society, 1977. doi:10.1109/SFCS.1977.24.

Uncrossed Multiflows and Applications to Disjoint Paths

Chandra Chekuri
Department of Computer Science, University of Illinois, Urbana-Champaign, IL, USA

Guyslain Naves
Aix-Marseille Univ, CNRS, LIS, Marseille, France

Joseph Poremba[1]
Department of Computer Science, University of British Columbia, Vancouver, Canada

F. Bruce Shepherd
Department of Computer Science, University of British Columbia, Vancouver, Canada

Abstract

A multiflow in a planar graph is *uncrossed* if the curves identified by its support paths do not cross in the plane. Recently, uncrossed flows have played a role in approximation algorithms for maximum disjoint paths in "fully-planar" instances, where the combined supply-plus-demand graph is planar. They are also used in algorithms to find low-congestion unsplittable flows for both fully-planar and single-source instances. For these two instance classes, any fractional multiflow can be converted into one that is uncrossed, which these algorithms then exploit to obtain their results.

We investigate the utility of uncrossed flow more generally and ask three key questions. First, are there other interesting planar multiflow instances that admit uncrossed flows (beyond fully-planar and single-source)? We answer affirmatively, demonstrating a new family of "pairwise-planar" instances whose fractional flows can be uncrossed. This family subsumes fully-planar but includes substantially more, such as (2-connected) fully-compliant series-parallel instances and some instances that have large clique demand graphs. Second, can we always round a fractional uncrossed flow to a "good" integral flow? We again answer positively. For maximization problems, we show any fractional uncrossed flow can be rounded to an integral flow with a constant fraction of its value. For congestion problems (where we must fully route all given demands), we give a rounding procedure that yields an integral multiflow with edge congestion 2. Consequently, we obtain constant-factor approximation algorithms for maximum disjoint paths and minimum congestion integer multiflow for pairwise-planar instances, and show such instances have a constant integral flow-multicut gap. Finally we ask, given an arbitrary planar instance, can we determine if there exists a congestion-1 uncrossed fractional flow (congestion setting) or find the maximum value uncrossed fractional flow (maximization setting)? For congestion, we show this problem is NP-hard, but finding uncrossed edge-disjoint paths is polytime solvable if the demands span a bounded number of faces. For maximization, we present a strong (almost-polynomial) inapproximability result.

2012 ACM Subject Classification Mathematics of computing → Network flows; Mathematics of computing → Approximation algorithms

Keywords and phrases Network Flows, Disjoint Paths, Planar Graphs, Crossing, Flow-Multicut Gap, Approximation Algorithms, Combinatorial Optimization

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.12

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2511.00254

Funding *Chandra Chekuri*: Supported in part by NSF grant CCF-2402667.
F. Bruce Shepherd: Supported by NSERC Discovery Grant RGPIN-2026-06972.

[1] Corresponding author

© Chandra Chekuri, Guyslain Naves, Joseph Poremba, and F. Bruce Shepherd;
licensed under Creative Commons License CC-BY 4.0



Figure 1 A fully-planar multiflow instance with two different multiflows routing its demands. Dashed edges represent demands, and coloured paths are support paths of the flow. The support paths on the left cross, while the multiflow on the right is uncrossed.

1 Introduction

In *multicommodity flow*, we are given an edge-capacitated *supply graph* (G, u) and a set of source-sink pairs called *demands* or *commodities*. The demands are represented by a *demand graph* H on the same node set as G. Unless otherwise specified, graphs are *undirected*. We seek to route flow for the demands simultaneously, while respecting the capacities of the supply graph. There are two flavours of the multiflow problem. In the *maximization* setting, the objective is to find a flow x that respects the capacities and maximizes the total sum of demands being routed. A feasible flow for (G, H, u) needs to respect the capacities, but has freedom in the quantities of each demand that it routes. In the *congestion* setting, we are given target amounts $d(s, t)$ for each $(s, t) \in E(H)$. To be feasible for (G, H, u, d), a flow must (in addition to respecting capacities) fully route this amount for each demand. The objective is to determine whether d can be satisfied in the capacities u, or to find the minimum scaling factor $\alpha \geq 1$ such that d can be satisfied in capacities αu.

Both fractional and integer versions of these problems are fundamental in combinatorial optimization (cf. [28]). Due to applications, the literature often emphasizes problems where the supply graph is planar [20]. The integer problems are hard even for planar supply graphs with 3 demand edges [33]. There exist, however, good approximation algorithms in so-called *fully-planar* instances where the joint supply-demand graph $G + H$ is planar[2]. Often approximation algorithms follow a "relax-and-round" approach: relax the integer problem to its fractional version, find a fractional solution with "nice properties", and then round the nice fractional solution to an integral solution.

A recurring "nice property" for planar multiflow is that the support paths (i.e. paths routing non-zero flow) do not cross in the plane. A multiflow satisfying this property is *uncrossed* (see Figure 1). It is known how to construct uncrossed multiflows for single-source (H is a star) and fully-planar instances, the latter using an algorithm of Matsumoto et al. [21]. This was recently leveraged to give constant-factor approximation algorithms for *maximum edge-disjoint paths* (MEDP) [15] and *maximum node-disjoint paths* (MNDP) [25, 24] for fully-planar instances. Very recently uncrossed flows were used to round fractional flows to *unsplittable* flows, routing the same demand quantities with small additive error, in both single-source [31] and fully-planar contexts [12]. In an unsplittable flow, each demand is routed on a single path.

[2] In this paper, if we say a multiflow instance is *planar* we only mean that the supply graph G is planar. If we mean $G + H$ is planar, we will say fully-planar.

In the above results, the uncrossed property of the fractional solution plays a key role, but the arguments are still specialized to the structure of (G, H). For example, for the fully-planar context, the uncrossed flow can be shown to be *laminar*: if one considers the family of cycles obtained by joining demand edges of H to their support paths in G, the result is a nested family of cycles in the plane. However, this same property does not hold for all instances that admit uncrossed flows. For instance, if G is a cycle and H consists of mutually crossing demands, then no paths within G cross, but the path-plus-demand cycles are not nested. Our aim is to expand on a theory of uncrossed flows - when can we find them, and do they always admit good rounding algorithms, even outside the laminar/single-source cases? We have three motivating questions.

▶ **Question 1.** *Do there exist other (interesting) families of (G, H) satisfying the following property: for all capacities u and demand values d, if (G, H, u, d) has a feasible (fractional) multiflow x, then (G, H, u, d) has a feasible multiflow y that is uncrossed?*

We say (G, H) is an *uncrossable layout* if it satisfies the conditions of Question 1. For instance, if (G, H) is fully-planar or H is single-source, then (G, H) is an uncrossable layout.

▶ **Question 2.** *Suppose we are given a feasible fractional multiflow x that is uncrossed (not assuming that the multiflow x is laminar). Can we round x to a "good" integer multiflow y? Specifically:*

1. *(Congestion Setting) Given feasible uncrossed fractional multiflow x for (G, H, u, d), can we compute an integer multiflow y feasible for $(G, H, \alpha u, d)$, for some constant α?*
2. *(Maximization Setting) Given feasible uncrossed fractional multiflow x for (G, H, u), can we compute an integer multiflow y feasible for (G, H, u) whose value is $1/\beta$ times that of x, for some constant β?*

The answer to Question 1 and both parts of Question 2 is: yes! First, we define a new class of (G, H) and prove it gives uncrossable layouts.

▶ **Theorem 1.** *Let G be a planar graph with a fixed embedding. Let H be such that for any pair of demands $h_1, h_2 \in E(H)$, h_1 and h_2 can be embedded inside (possibly different) faces of G so they do not cross each other. Then (G, H) is an uncrossable layout.*

We say any (G, H) satisfying this property is *pairwise-planar* (see Figure 6 in Section 3 for a depiction). The pairwise-planar class includes fully-planar as a subset, but includes substantially more. For instance, it contains layouts where $G + H$ is highly non-planar (and thus, do not admit laminar flows). It also includes the 2-connected *fully-compliant*[3] family introduced in [7], which were not previously known to give uncrossed layouts to our knowledge. Pairwise-planar also differs from previous uncrossable layouts in an important structural sense. Namely, fully-planar and single-source (G, H) are *cut-sufficient* (essentially, they have a Max-Flow Min-Cut theorem in the congestion setting), while some pairwise-planar (G, H) are not.

As for Question 2, we prove the following results. We begin with the maximization setting.

▶ **Theorem 2.** *Let (G, H, u) be a planar maximization instance where u is integral. Let f be a feasible fractional flow that is uncrossed. There exists a feasible integral flow f' achieving an $\Omega(1)$ fraction of the value of f. The integral flow can be computed in polynomial time.*

[3] (G, H) is fully-compliant if G is series-parallel and $G + h$ is series-parallel for every $h \in E(H)$.

We prove this result by establishing an "approximate integer decomposition property", exploiting colouring results on a restricted class of string graphs (path intersection graphs). This integer decomposition also implies $\Omega(1)$ rounding holds for the *weighted* maximization version, where each $(s,t) \in E(H)$ has an associated weight $w(s,t)$, and a flow achieves $w(s,t)$ value for each unit of (s,t)-flow that it routes. Additionally, this result extends to *node-capacitated* multiflows (i.e. u is a capacity function on nodes rather than edges).

For the congestion setting, we give a factor 2 rounding for converting a fractional uncrossed multiflow into an integral multiflow. Moreover, for an extra additive penalty we can round to an unsplittable flow.

▶ **Theorem 3.** *Let (G, H, u, d) be a planar congestion instance where u, d are integral. Let f be a feasible fractional multiflow that is strongly uncrossed*[4]*. Then there exists both:*

1. *an integral multiflow f' that is feasible for (G, H, u', d) where $u'(e) \leq 2u(e)$ for each $e \in E(G)$, and*
2. *an unsplittable multiflow f'' that is feasible for (G, H, u'', d) where $u''(e) < 2u(e) + d_{\max}$ for every $e \in E(G)$, where $d_{\max} = \max_{h \in E(H)} d(h)$.*

The integral and unsplittable multiflows can be computed in polynomial time.

Using Theorems 1, 2, and 3, we obtain constant-factor approximation algorithms for integer multiflow problems with pairwise-planar inputs.

▶ **Corollary 4.** *There exist polynomial time constant-factor approximation algorithms for maximum edge-disjoint paths and maximum node-disjoint paths (and more generally, maximum integer multiflow), as well as minimum congestion integer multiflow, for inputs where (G, H) is pairwise-planar.*

Another interesting corollary concerns the *integral flow-multicut gap*, which is the ratio of the minimum capacity of a multicut (a set $F \subseteq E(G)$ whose removal destroys all paths connecting demands of H) to the maximum value of an *integral* multiflow. Using their flow rounding algorithms, [17] [15] show the gap is constant for fully-planar (G, H), but the gap can be as large as $\Theta(|E(H)|)$ even for planar supply graphs [14]. Combining our results with a theorem of [30], we obtain a constant gap for pairwise-planar instances.

▶ **Corollary 5.** *For any maximization instance (G, H, u) where (G, H) is pairwise-planar, the minimum capacity of a multicut is at most $O(1)$ times the maximum value of an integral multiflow.*

The third key question requires slightly more motivation. If (G, H) is not an uncrossable layout, it only means the structure of (G, H) alone is insufficient to guarantee uncrossed solutions. However for specific capacities u and demands d we may still be able to find an uncrossed flow and round it. Thus it would be nice to have algorithms to determine whether (in the congestion setting) we can find a feasible fractional flow that is uncrossed, or (in the maximization setting) find the largest feasible flow that is uncrossed.

▶ **Question 3.** *Suppose we are given an* arbitrary *planar multiflow instance. Can we solve the fractional multiflow problem if we constrain the feasible space to only allow uncrossed solutions? Specifically:*

[4] We defer the more technical definition of strongly uncrossed to Section 2.4. Importantly, the uncrossable layouts mentioned in this paper, including pairwise-planar, admit strongly uncrossed flows.

1. *(Congestion Setting) Given* (G, H, u, d)*, can we determine whether there exists a feasible fractional multiflow that is uncrossed?*
2. *(Maximization Setting) Given* (G, H, u)*, can we compute a feasible fractional uncrossed multiflow whose value is (approximately) maximum amongst feasible uncrossed solutions (*maximum uncrossed multiflow*)?*

These questions concern *fractional* multiflows and may feel like linear optimization problems. However the requirement of uncrossed flow paths makes these computationally much more challenging. Interestingly, the positive result of Theorem 2 can be combined with results of [8, 9] to establish strong almost-polynomial inapproximability for maximum uncrossed fractional multiflow (for both edge and node capacities).

▶ **Theorem 6.** *For planar* G*, maximum uncrossed fractional multiflow is not approximable to within a factor of* $2^{\Omega(\log^{1-\epsilon} n)}$ *for any* $\epsilon > 0$ *assuming that* $NP \not\subseteq DTIME(n^{polylog\ n})$*. Moreover, there is no* $n^{O\left(\frac{1}{(\log\log n)^2}\right)}$*-approximation polynomial-time algorithm assuming that for some constant* $\delta > 0$*,* $NP \not\subseteq DTIME(2^{n^\delta})$*. These results hold even when* G *is maximum degree* 3 *(or even* G *is a wall graph) and the capacities are unit.*

For the congestion setting, we prove the problem posed by Question 3 is NP-complete.

▶ **Theorem 7.** *It is NP-complete to determine whether planar congestion instance* (G, H, u, d) *has a feasible fractional uncrossed multiflow, even with unit-valued capacities and demands.*

We remark that since this problem is not a linear program anymore, it is not even obvious that the problem is in NP, however we establish this as well. On the positive side, we show that finding uncrossed edge-disjoint paths can be (polytime) reduced to a node-disjoint paths instance. Hence we may find uncrossed solutions if $E(H)$ is incident with a bounded number of faces using a result of Schrijver [27].

▶ **Theorem 8.** *For instances where* G *is planar, the edges of* H *are incident with a bounded number of faces, and the capacities and demands are unit-valued, there is a polynomial time algorithm to find a feasible integral uncrossed multiflow for* (G, H, u, d) *or declare none exists.*

1.1 Organization of the Paper

Section 2 lays the groundwork for the theory of uncrossed flows, including detailed definitions. Section 3 discusses pairwise-planar instances, proves they admit uncrossed flows, and shows a constant integral flow-multicut gap. Section 4 covers our maximization results, both rounding (Theorem 2) and inapproximability (Theorem 6). The rounding result for congestion (Theorem 3) appears in Section 5. This manuscript includes the important aspects of each proof, though occasionally some details are deferred to the full version. The NP-completeness result (Theorem 7) and the integral uncrossed flow algorithm for a bounded number of H-incident faces (Theorem 8) are deferred to the full version.

1.2 Related Work

Uncrossed flows have played a role in previous routing algorithms in planar graphs, such as Schrijver's Homotopy Method [26] for node-disjoint paths. Most relevant are recent algorithms for maximum disjoint paths in fully-planar instances ($G + H$ planar). In [17, 15], constant-factor approximation algorithms are given for maximum edge-disjoint paths, then extended to the node-disjoint setting in [25, 24]. These algorithms glue uncrossed flow paths to their respective demand edges to obtain an uncrossed (laminar) family of cycles in a

planar graph. Our rounding algorithms are agnostic in the sense that they just operate on uncrossed flow paths, not assuming that demands lie within a face or that demands of a common face are non-crossing (indeed for our pairwise-planar instances, it is even possible for demands within a face to form a clique).

Uncrossed flows for fully-planar instances are typically constructed by a primal algorithm [21] that computes an uncrossed flow from scratch. We build on a different approach, taking a feasible flow (with many crossings) and iteratively decreasing total crossing (used for example in [18] to find cycles in higher-genus surfaces that cross at most once). We discuss obstacles faced in extending this to pairwise-planar in Section 3.

In [15] the initial flow is first rounded to a half-integral flow (using linear programming theory such as total unimodularity), then this half-integral flow is rounded to an integral flow using a 4-colouring of the (planar) intersection graph of the cycles. Extending to the node-disjoint setting, [25] uses a different argument, iteratively constructing a high-value solution by showing one can always find an "efficient" cycle that only intersects the other cycles on a small set of its nodes. Our rounding technique for Theorem 2 instead uses an approach based on establishing an "approximate integer decomposition property" [3, 5]. A key ingredient comes from looking at a subset of *string graphs*, which are the intersection graphs from curves in the plane. Specifically, we are interested in those that arise from curves that do not cross, studied for example in [11], and for which an important colouring result was proven by Fox and Pach [13][5].

The fully-planar maximum disjoint paths algorithms were extended [25, 18] to uncrossable families of cycles in higher-genus surfaces, yielding approximation algorithms for maximum disjoint paths when $G + H$ is embedded in a surface of genus g; the approximation ratio is polynomial in the genus g. By contrast our work extends uncrossing techniques to a broader family of multiflow instances (pairwise-planar) while remaining *in the plane*. Some pairwise-planar instances have $G + H$ of arbitrarily high genus (e.g. H contains a large clique), but we show nevertheless that they admit uncrossed flows in the plane in order to achieve constant-factor approximations. Results in [25] use the context of uncrossed cycle families, as defined by Goemans and Williamson [16], which is a combinatorial rather than topological definition of crossing.

The works of [17], [15] use their rounding algorithms to establish that the integral flow-multicut gap of fully-planar instances is constant. The integral flow-multicut gap can be as large as $\Theta(|E(H)|)$ for general planar G [14], so further restrictions on (G, H) are needed to establish constant bounds. Other classes known to give constant integral flow-multicut gaps include: when G is a tree [14], when $G + H$ is series-parallel [10] (note that these are less general than fully-compliant), when G has fixed cyclomatic number [1], when $G + H$ has fixed genus [18], and when G has fixed tree-width [4]. A common ingredient in these results is a bound on the *fractional* flow-multicut gap for $K_{r,r}$-minor-free G by [30, 19].

Other works using uncrossed flow come from the study of unsplittable flow in the congestion model (where specified demand quantities must be fully routed). For directed single-source planar instances (H is a star) [31], it is shown how to convert a feasible fractional uncrossed flow into an unsplittable flow that exceeds capacities by a small amount and has no increase in cost. For (undirected) fully-planar instances, [12] shows how to convert a laminar flow into an unsplittable flow y with small capacity violation (no cost guarantee).

[5] This manuscript is unpublished, but the result is mentioned in [32] and their writeup was shared with us [23].

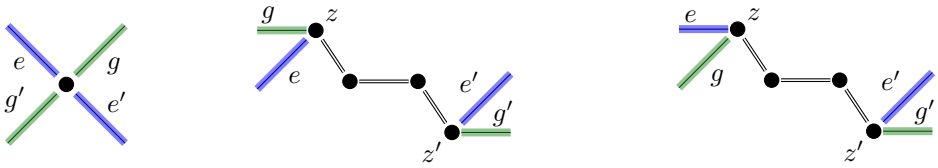


Figure 2 (Left) Two edge-disjoint paths that cross at a common node. (Center) Two paths that cross at a common subpath. (Right) Two paths that share a common subpath, but are uncrossed. Uncoloured edges are shared between the two paths.

2 Uncrossed Multiflows

2.1 Multiflow Preliminaries

All graphs are undirected in this paper. Consider a supply graph $G = (V, E(G))$ and demand graph $H = (V, E(H))$. Endpoints of edges in H are *terminals*, and the edges themselves are *demands* or *commodities*. Together, (G, H) define a *layout*[6]. For nodes $s, t \in V$, let $\mathcal{P}_{st}$ denote the set of st-paths. By *path* we always mean simple path. A *multiflow* is a function $x : \mathcal{P} \to \mathbb{R}_{\geq 0}$, where $\mathcal{P} = \bigcup_{(s,t)\in E(H)} \mathcal{P}_{st}$. Multiflows are fractional unless explicitly stated as integral. A multiflow can also be viewed as a collection of single-commodity flows $(x^{st} : (s, t) \in E(H))$. For $e \in E(G)$, we let $x(e) = \sum_{P\in\mathcal{P}:e\in P} x(P)$ (the amount of flow on e). For $(s, t) \in E(H)$ we let $|x^{st}| = \sum_{P\in\mathcal{P}_{st}} x(P)$ (the amount of demand (s, t) that is routed).

A *maximization instance* has the form (G, H, u), where $u : E(G) \to \mathbb{R}_{\geq 0}$ are *capacities*. A multiflow x is *feasible* for the instance if for every $e \in E(G)$, $x(e) \leq u(e)$. The objective is to maximize the value of a feasible multiflow, where the value of a multiflow x is $|x| = \sum_{P\in\mathcal{P}} x(P) = \sum_{(s,t)\in E(H)} |x^{st}|$. If $u = \vec{1}$ and we restrict to integral x, this problem is the *maximum edge-disjoint paths* problem (MEDP). A *weighted* maximization instance also comes equipped with per-unit profits $w : E(H) \to \mathbb{R}_{\geq 0}$, in which case the value of x is instead defined $|x|_w = \sum_{(s,t)\in E(H)} w(s, t)|x^{st}|$.

A *congestion instance* has the form (G, H, u, d), where $u : E(G) \to \mathbb{R}_{\geq 0}$ are capacities and $d : E(H) \to \mathbb{R}_{\geq 0}$ are *demand values*. A multiflow x is *feasible* for the instance if for every $e \in E(G)$ we have $x(e) \leq u(e)$ and for every $(s, t) \in E(H)$ we have $|x^{st}| = d(s, t)$. As a decision problem, the goal is to determine whether (G, H, u, d) has a feasible multiflow. As an optimization problem, the objective is to find the minimum $\alpha \geq 1$ such that $(G, H, \alpha u, d)$ has a feasible multiflow.

In either setting, we say an instance is *planar* if G is planar. We emphasize that $G + H$ is not necessarily planar. We always assume a fixed embedding of planar graph G.

2.2 Uncrossed Paths and Multiflows

First, we define what it means for paths to cross. Edge-disjoint paths can cross at a common node, but in general we need to consider common subpaths (see Figure 2).

▶ **Definition 9** (Crossing). *Let Z be a maximal shared subpath of paths P and Q that does not include any of their endpoints. Say the ends of Z are z and z'. The edges of $(P \cup Q) \setminus Z$ incident with z or z' have a clockwise ordering about Z in the plane. The paths* cross *at*

[6] Sometimes in the literature, (G, H) is called a *multiflow topology*. We avoid this term to avoid confusion with the topology of the plane.

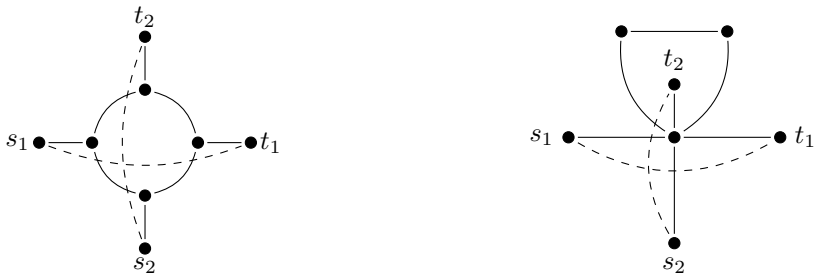


Figure 3 Problematic instances for uncrossability. Capacities and demand values are unit. (Left) A congestion instance with a feasible multiflow but no uncrossed multiflow. (Right) The *Keyhole Instance*, a congestion instance with a feasible uncrossed trail-multiflow but no uncrossed path-multiflow.

Z if the edges of P and Q alternate clockwise about Z. That is, if $P = P_1, e, Z, e', P_2$ and $Q = Q_1, g, Z, g', Q_2$, the edges occur e, g, e', g' or e, g', e', g clockwise about Z. Two paths are uncrossed *if they do not cross at any such subpath*[7].

The *support paths* of a multiflow x are the paths with $x(P) > 0$. The following is the natural extension of "uncrossed" to the multiflow setting (though we discuss a stronger condition in Section 2.4).

▶ **Definition 10** (Uncrossed Multiflow). *A multiflow is* uncrossed *if its support paths are pairwise uncrossed.*

We are interested in when a feasible multiflow x can be converted into an uncrossed one that routes the same demands. Our discussion here focuses on the congestion setting, but still applies to the maximization setting (since we can define target demand values d equal to whatever quantities x routes). As shown in Figure 3, even if (G, H, u, d) is feasible, it may not have an uncrossed feasible multiflow[8]. However several classes do admit uncrossed solutions. We say (G, H) is an *uncrossable layout* if for all u, d, if (G, H, u, d) has a feasible multiflow, then (G, H, u, d) has a feasible uncrossed multiflow. Matsumoto et al. [21] give an algorithm that, given a feasible fully-planar ($G + H$ planar) instance, produces a feasible uncrossed flow. Thus fully-planar layouts are uncrossable. It is also well-known that single-source layouts (H is a star) are uncrossable. Another uncrossable layout is given when G is a cycle with parallel edges and H is arbitrary, which we call *ring* layouts. In Section 3 we introduce *pairwise-planar* layouts and prove they are uncrossable.

2.3 Another View on Crossing: Parallelizations

There are more intuitive ways to think about uncrossed paths that we frequently use.

[7] We emphasize that paths can only cross at maximal common subpaths that do *not* contain their endpoints. Both paths must have edges before and after the common subpath in order to cross. For example, if P contains an endpoint of Q and Z is the maximal common subpath containing that endpoint, then P and Q do not cross at Z.

[8] In fact, it is not always possible to find an uncrossed flow with bounded congestion. One may modify the canonical grid instance [14] so it has a half-integral routing of demands, but any uncrossed multiflow incurs polynomial congestion.



Figure 4 (Left) An uncrossed parallelization of the uncrossed paths from Figure 2. (Right) Disk-expansion of the node z after uncrossed parallelization.

Widened Graph. Suppose we "widen" G, making its nodes into small disks and its edges into small channels. A family of pairwise uncrossed paths can have its paths drawn as curves simultaneously, each curve following the node disks and edge channels corresponding to its path, so that the curves are disjoint.

Parallelizations. To avoid dealing with the formal topology of widened graphs, we use an equivalent notion based on "separating" the paths at each edge. Consider a family of paths $\mathcal{P}$ and for each $e \in E(G)$ let $\mathcal{P}_e = \{P \in \mathcal{P} : e \in E(P)\}$. Let G' be obtained by replacing each e with $|\mathcal{P}_e|$ parallel copies of itself. Intuitively, the parallel edges represent the curves within the channel of e.

Now, for each $e \in E(G)$ we decide on some one-to-one mapping ϕ_e from $\mathcal{P}_e$ to the parallel class of e in G'. For each $P \in \mathcal{P}$, we obtain a path P_ϕ in G' whose edges are $\{\phi_e(P) : e \in P\}$. This induces a family of edge-disjoint paths $\mathcal{P}_\phi$ in G'. We call $\mathcal{P}_\phi$ a *parallelization* of $\mathcal{P}$. If we (arbitrarily) orient each $e \in E(G)$, it gives a linear ordering $\leq_e$ of $\mathcal{P}_e$. Different choices for $(\phi_e : e \in E(G))$ yield different parallelizations. The family $\mathcal{P}$ is pairwise uncrossed if and only if there is some parallelization that is pairwise uncrossed at every node of G' (rather than needing to consider longer shared subpaths). Using parallelizations helps us simplify subsequent definitions by assuming paths are edge-disjoint. Given a collection of paths, one can compute in polynomial time an uncrossed parallelization or find a crossing pair of paths.

Disk-Expansion. After parallelization, we sometimes go a step closer to the widened graph idea by "expanding" a node v. We draw a small disk with v at its centre (small enough that its boundary intersects only $\delta(v)$, and hits each edge once). Where an edge copy intersects the boundary, we place a new node. We terminate the edges at these new nodes. For an uncrossed parallelization, it is then possible to (simultaneously) draw edges in the open interior of the disk between nodes that correspond to consecutive edges of the same path, such that the drawing is planar. An example is shown in Figure 4.

2.4 Strongly Uncrossed Multiflows

Consider an uncrossed multiflow. For any $(s,t) \in E(H)$, its support paths have a clockwise ordering about s (and t) given by the parallelization. We call the regions of the plane bounded between two consecutive paths *sectors* for this commodity (defined more formally in Section 5).

The proof of Theorem 3 uses that sectors of different commodities are non-crossing as sets - that is, we want that if $A, B \subseteq \mathbb{R}^2$ are sectors of different commodities, then either $A \cap B = \emptyset$ or one contains the other. One might think this non-crossing property follows

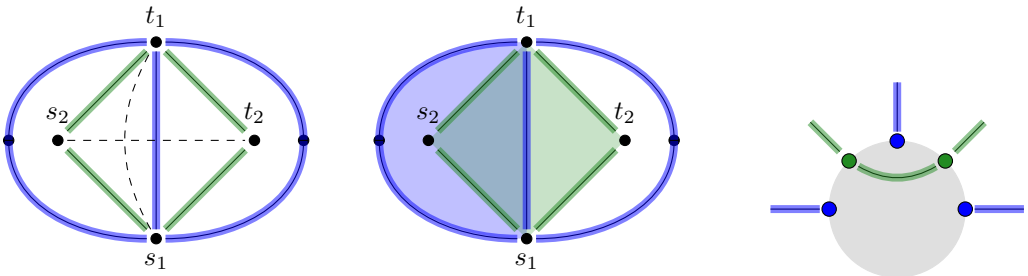


Figure 5 (Left) An uncrossed multiflow. The blue paths are for (s_1, t_1) and the green paths are for (s_2, t_2). No pair of support paths cross. (Center) A sector from the s_1t_1-flow (blue shaded region) that crosses a sector from the s_2t_2-flow (green shaded region). (Right) The disk-expansion of s_1. The green path separates the blue nodes in the disk. There is a quasicrossing at s_1.

from Definition 10, however Figure 5 shows a counterexample[9]. In the figure, no pair of flow paths cross, but the green s_2t_2-paths "slip between" the blue s_1t_1-paths to move between two distinct sectors. We define *strongly uncrossed* multiflows by imposing stricter restrictions on the support paths of different commodities at their terminals. Many layout classes admit strongly uncrossed flows, including fully-planar, single-source, and rings, as well as pairwise-planar.

▶ **Definition 11** (Quasicrossing, Strongly Uncrossed). *Let f be a multiflow with a fixed parallelization (so we assume edge-disjoint support paths). Consider a node z and the clockwise ordering of edges $e_1, \ldots, e_k$ about z. For a commodity h, a* type-1 (h, z)-interval *is $[i, j]$ (where $i < j$) such that e_i, e_j are consecutive edges of a single support path for h that transits through z. A* type-2 (h, z)-interval *is $[i, j]$ such that e_i, e_j both terminate (different) support paths for h at z. Note that two paths cross at z if and only if their corresponding type-1 intervals cross*[10]*. We say a* quasicrossing *occurs at z if there exists an (h, z)-interval I and an (h', z)-interval I' that cross where $h \neq h'$ and at least one of the intervals is type-2. If there are no crossings and no quasicrossings at any node, we say f is* strongly uncrossed.

2.5 Walk-Multiflows

It is standard for flows to only route on (simple) paths, since a flow walk could just be short-circuited into a flow path. Disturbingly, this operation is not benign in the uncrossed flow setting. The *Keyhole Instance* of Figure 3 is an example that requires the use of a non-simple flow trail to obtain an uncrossed solution. Since we would like to understand which instances have uncrossed solutions, we must consider *trail-multiflows* and *walk-multiflows* in addition to the usual *path-multiflows*. A *trail* can repeat nodes but not edges, while a *walk* can repeat both. Ultimately we can show our rounding results (Theorem 2, Theorem 3) and hardness results (Theorem 6, Theorem 7) presented earlier hold for all three models. The necessary modifications are minor and we defer the details to the full version.

[9] The desired non-crossing sector property does hold for uncrossed flows in *leaf instances*, where terminals all have degree 1. Moving terminals to leaves is a standard pre-processing operation, but is not safe for uncrossed flows - it may turn an uncrossable layout into one that is not. For example, ring instances admit uncrossed flows but Figure 3 (left) does not, even though it is obtained by adding leaves to a ring.

[10] Two intervals $[i, j], [i', j']$ cross if $i < i' < j < j'$ or $i' < i < j' < j$.

3 Uncrossed Flows in Pairwise-Planar Instances

Here we introduce a new class of multiflow layouts and prove they are uncrossable. We assume a fixed embedding of G. We say $(s,t) \in E(H)$ *fits* face F of G if both s,t lie in F (i.e. the demand could be drawn within the face). A layout (G,H) is *facial* if each (s,t) fits at least one face of G (but demands fitting a common face may still cross each other).

▶ **Definition 12.** *We say (G,H) is* pairwise-planar *if (G,H) is facial, and moreover for any pair $h_1, h_2 \in E(H)$, h_1 and h_2 can be embedded inside (possibly different) faces of G so they do not cross each other.*

Unlike fully-planar, we do not require simultaneous uncrossed embedding of all demands. This matters because a demand edge may fit multiple faces (a *multi-faced* demand). Figure 6 shows an example. However we emphasize that the embedding of G is fixed and does not change as we embed different pairs of $E(H)$.

Clearly pairwise-planar generalizes fully-planar, but how substantial is this generalization? We briefly argue it is a much richer class. First, it is easy to construct examples where H contains arbitrarily large cliques, meaning H and $G+H$ have arbitrarily large genus (though G remains planar). It also generalizes rings (G is a cycle with parallel edges, H is arbitrary). This is worth remarking because fully-planar does not capture ring layouts. More broadly, pairwise-planar captures what are known as *fully-compliant* layouts (under mild connectivity assumptions). Introduced in [7], in such layouts G is series-parallel and every $h \in E(H)$ satisfies that $G+h$ is series-parallel. In the full version, we argue the following.

▶ **Proposition 13.** *If (G,H) is fully-compliant and G is 2-connected, then (G,H) is pairwise-planar for any planar embedding of G.*

Pairwise-planar extends further than fully-planar and fully-compliant in another fundamental way through a property known as *cut-sufficiency*. For a multiflow instance (G,H,u,d), we say the *cut condition* holds if for every $S \subseteq V$, $u(\delta_G(S)) \geq d(\delta_H(S))$, where $\delta(S)$ denotes the edges with one end in S and one end in $V \setminus S$. The cut condition is necessary, but not in general sufficient, for the existence of a feasible (fractional) multiflow. A layout (G,H) is called cut-sufficient if, for all capacities and demands (u,d), the cut condition is sufficient for the existence of a feasible multiflow. The Max-Flow Min-Cut Theorem can be equivalently stated as: single-source (G,H) are cut-sufficient. Understanding the relationship between the cut condition and feasibility is a fundamental question in network flows, and cut-sufficient instances tend to admit stronger integrality properties and approximation algorithms. A result of Seymour [29] shows that fully-planar (G,H) are cut-sufficient, and Chekuri et al. [7] show that fully-compliant instances are cut-sufficient. However, pairwise-planar layouts need not be cut-sufficient. The quintessential examples of series-parallel layouts that are not cut-sufficient are the *odd spindles*[11] [6, 2], but these are pairwise-planar.

3.1 Uncrossing Procedure

In this section we show why feasible pairwise-planar instances have uncrossed flows. We focus on congestion instances (G,H,u,d) (where a feasible flow must satisfy all demands), since given a feasible flow x for a maximization instance (G,H,u), one can define demand

[11] In a *spindle*, $G = K_{2,p}$ where $p \geq 3$ and H consists of a simple p-cycle between the nodes in the larger side of the partition and an edge between the two nodes in the other side. A spindle is *odd* if p is odd. In some sense, the odd spindles characterize non-cut-sufficiency for series-parallel layouts, see [2].

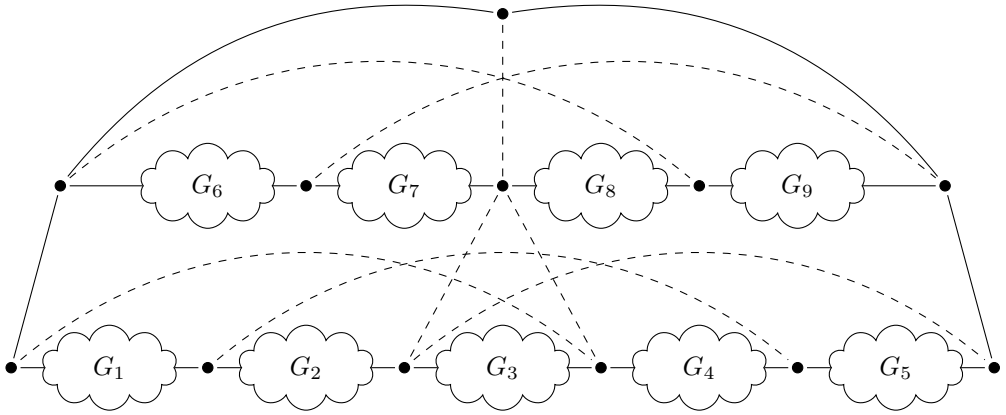


Figure 6 A pairwise-planar layout. Each G_i is a connected planar graph. The demand edges cannot be simultaneously embedded so that $G + H$ is planar ($G + H$ contains a $K_{3,3}$-minor and a K_5-minor). However, taking advantage of multi-faced demands, any *single pair* of demands can be embedded without crossing. One could make the layout more complicated by putting demands inside the G_i's following similar patterns.

values $d(s,t) = |x^{st}|$ and apply the congestion result. We present an inductive proof that a feasible uncrossed flow exists. It is straightforward to convert this into an algorithm, with one catch for polynomial time implementation that we explain at the end.

For fully-planar instances, [15, 12] use an algorithm by Matsumoto et al. [21] that builds an uncrossed flow from scratch starting from the zero flow. Our procedure has more in common with [18] (which concerns finding cycles in higher-genus surfaces that cross at most once) in that we begin with a (potentially highly crossing) feasible flow x and "uncross" it bit-by-bit. The following is an important lemma.

▶ **Lemma 14.** *Let (G, H, u, d) be a planar congestion instance that is feasible. There exists a feasible flow f^* such that every pair of support paths crosses at most once, and any pair of support paths that share a terminal do not cross.*

We say a flow satisfying this property is *almost-uncrossed*. Proving this lemma formally is somewhat tedious, but such arguments have appeared in the literature before - the essence of the argument is in Lemma 1 of [22] and a similar method is used in [18]. Loosely, the argument goes as follows. Begin with a feasible flow x. One can show that if a pair of support paths cross twice (or cross once but share a terminal), then one can re-route flow on their edges to decrease the "amount of crossing" between the two paths ("uncrossing" the two paths) and argue that this re-routing does not increase crossing with other support paths. Thus the total amount of crossing decreases, and we iterate until x is uncrossed. For completeness we give a careful proof of this lemma (and precisely define "amount of crossing") in the full version.

Lemma 14 immediately implies single-source layouts are uncrossable. With a small trick it also proves fully-planar layouts are uncrossable: augment the instance by adding leaves. That is, for each $h = (s,t) \in E(H)$, add nodes s_h, t_h into the interior of the face where h is embedded, add supply edges $(s_h, s), (t_h, t)$, move the endpoints of the demand h to (s_h, t_h), and extend the flow paths appropriately to route this instance. At first, adding leaves may actually increase the amount of crossing. However, since the demands do not cross, it can be shown that every pair of support paths that cross must cross at least twice. Then Lemma 14 allows one to conclude there exists a feasible uncrossed flow in the augmented instance. The leaves can then be contracted away without introducing any new crossing.

Extending this trick to pairwise-planar instances is not immediate. After adding leaves, it may not be the case that support paths that cross must cross at least twice. We cannot "commit" a demand to a particular face up front. Consider if a demand h fits into two different faces F_1 and F_2, but there are demands h_1 and h_2 that fit those respective faces and cross h in those faces. If we add leaves into F_1, then we can "uncross" support paths for h and h_2, however we cannot uncross support paths between h and h_1, and vice versa.

We take the following approach. Begin with a feasible flow z that we want to uncross. We temporarily "commit" each demand to one of the faces it fits (if multi-faced, choose arbitrarily). We accept for now that flow paths of demands committed to the same face may cross, but compute a flow x so that demands committed to different faces do not cross. In fact we aim for a more strict condition on x: we show how to find x so that, if we define x^F as the multiflow corresponding to demands committed to face F, the x^F's are "uncoupled" from each other. Here, "uncoupled" is a structural property (defined precisely below) which guarantees that if we can uncross each x^F separately, regardless of how we do so, we can still safely glue the resulting flows back together without introducing crossing. Then we can just focus on uncrossing each x^F inductively/recursively. We now make this "committing" process and "uncoupled" property precise.

▶ **Definition 15** (Facial Decomposition). *Let (G, H, u, d) be a facial multiflow instance. Let $\mathcal{F}$ be the set of faces of G. A* facial decomposition *is a collection $(H_F : F \in \mathcal{F})$ of subgraphs of H indexed by F such that:*

- *$(E(H_F) : F \in \mathcal{F})$ forms a partition of $E(H)$, and*
- *for every face $F \in \mathcal{F}$, every demand $h \in E(H_F)$ fits F (but cannot necessarily all be embedded simultaneously in F without crossing).*

For a facial decomposition $(H_F : F \in \mathcal{F})$ and a feasible flow x, we let x^F be the portion of x used to route H_F. Note that if $h \in E(H)$ is multi-faced, the facial decomposition still only puts h into one of the H_F's. Now we define the "uncoupled" condition and show that we can find a flow satisfying this property. We let $\mathrm{supp}(x^F)$ denote the set of support paths of x^F, and let $G[\mathrm{supp}(x^F)]$ denote the graph induced by the edges in those paths.

▶ **Definition 16** (Nested Supports Property). *Let (G, H, u, d) be a facial multiflow instance, and let $(H_F : F \in \mathcal{F})$ be a facial decomposition. For a flow x and a face F, we define the* stellated support graph *$G[x, F]$ as follows: begin with the support graph $G[\mathrm{supp}(x^F)]$, add a node v_F into the interior of F, and then for each demand $(s, t) \in E(H_F)$ add edges (s, v_F) and (t, v_F), embedded (without crossing) in the face F. We say x satisfies the* nested supports property *if for every pair of distinct faces $F_1, F_2 \in \mathcal{F}$, the graph $G[x, F_1]$ is entirely contained within a single face of the graph $G[x, F_2]$, and vice versa.*

▶ **Lemma 17.** *Let (G, H, u, d) be a facial multiflow instance that is feasible. Let $(H_F : F \in \mathcal{F})$ be a facial decomposition. There exists a feasible flow x satisfying the nested supports property.*

The proof of this lemma is slightly tricky to make rigorous, so we include only a sketch here and defer details to the full version.

Proof Sketch of Lemma 17. We augment the instance by adding leaves into the faces: for every face F and demand $h = (s, t) \in E(H_F)$, add nodes s_h, t_h into the face F and supply edges $(s_h, s), (t_h, t)$ of capacity equal to $d(h)$, and move the demand h from (s, t) to (s_h, t_h). We keep the same facial decomposition. We now invoke Lemma 14 to obtain a feasible flow x that is almost-uncrossed for this new instance. It is not hard to show that for support

paths P_1, P_2 associated with distinct faces, if we consider the embedded cycles $C(P_1), C(P_2)$ obtained by joining the endpoints of the paths with their respective demand edges, that these cycles do not cross (though cycles associated with paths of the same face may cross).

Now, for each face F, we let G_F be the union of $G[\text{supp}(x^F)]$ and all supply edges incident with the face F itself (regardless of whether they are in the support). It is possible to argue that for distinct faces F, F', G_F must be contained in a single face of $G_{F'}$, and vice versa. Once this property is established, we argue the nested supports property holds for flow x. ◀

Given such a flow, we show that we can focus on uncrossing each face's flow independently.

▶ **Lemma 18.** *Let (G, H, u, d) be a facial multiflow instance. Let $(H_F : F \in \mathcal{F})$ be a facial decomposition, and let x be a feasible flow satisfying the nested supports property. For every $F \in \mathcal{F}$, let u_F be the capacity vector defined by $u_F(e) = x^F(e)$ and let d_F be the demand vector d restricted to H_F. If for every $F \in \mathcal{F}$, the instance (G, H_F, u_F, d_F) has a feasible uncrossed solution y^F, then $y = \sum_{F \in \mathcal{F}} y^F$ is a feasible uncrossed solution for (G, H, u, d).*

Proof. The feasibility of y is clear from the construction of the subproblems. Also observe that, by construction, y still satisfies the nested supports property (since $G[y, F]$ is a subgraph of $G[x, F]$ for each face F of G).

We prove that y is uncrossed assuming each y^F is uncrossed. It remains to check for distinct faces F_1, F_2 and $P_1 \in \text{supp}(y^{F_1}), P_2 \in \text{supp}(y^{F_2})$, that P_1 and P_2 do not cross.

Consider two distinct faces F_1, F_2. Let $P_1 \in \text{supp}(y^{F_1}), P_2 \in \text{supp}(y^{F_2})$. Let $h_1 = (s_1, t_1), h_2 = (s_2, t_2)$ be the demands routed by P_1, P_2 respectively. Suppose P_1 and P_2 cross, for the sake of contradiction. Consider the (simple) cycle C in $G[y, F]$ obtained as $C = P_1 \bullet (s_1, v_{F_1}) \bullet (v_{F_1}, t_1)$, where $\bullet$ denotes concatenation. Since P_1 and P_2 cross, P_2 has some segment in the open exterior of C and some segment in the open interior of C. However, since C is a simple cycle, these two segments of P_2 must lie in distinct faces of $G[y, F]$, contradicting that y satisfies the nested supports property. ◀

Using Lemma 18, we are able to reduce to finding a feasible uncrossed flow for each instance (G, H_F, u_F, d_F) in isolation. No matter how we do this, we can safely combine the flows using the lemma. Importantly, to tackle the instance (G, H_F, u_F, d_F), we are allowed to re-commit multi-faced demands that were originally committed to F to other faces.

▶ **Theorem 19.** *Let (G, H) be pairwise-planar and let (G, H, u, d) be a congestion instance. If the instance is feasible, there exists a feasible uncrossed flow.*

Proof. We prove the result by induction on $|E(H)|$. If $|E(H)| = 1$ then the instance is single-source and the result follows, so suppose $|E(H)| \geq 2$. Let $\mathcal{H} = (H_F : F \in \mathcal{F})$ be an arbitrary facial decomposition.

We first consider the case that at least two faces F have that H_F is non-empty. We obtain a feasible flow x for (G, H, u, d) that satisfies the nested supports property for $\mathcal{H}$ by way of Lemma 17. For each face F, let u_F, d_F be defined as in the statement of Lemma 18. Each instance (G, H_F, u_F, d_F) is feasible via the flow x^F. Additionally, since there are at least two faces F such that H_F is non-empty, we have $|E(H_F)| < |E(H)|$ for every face F. Hence by induction, there is a feasible uncrossed flow y^F for each instance (G, H_F, u_F, d_F). Then by Lemma 18, the sum of these flows is a feasible uncrossed flow for (G, H, u, d).

Now, we consider the case where H_F is empty for all faces F except one, call it F^*. That is, $E(H) = E(H_{F^*})$, and in particular every demand fits F^*. If none of the demands cross within F^*, then the layout is fully-planar and we are done. Otherwise, there exist $h, h' \in E(H)$ that cross when drawn in the face F^*. However, since (G, H) is pairwise-planar,

there exists another face $F' \neq F^*$ such that at least one of these two demands, without loss of generality say h', also fits F'. Now we define a facial decomposition $\widehat{\mathcal{H}} = (\widehat{H}_F : F \in \mathcal{F})$ as follows. We define $E(\widehat{H}_{F^*}) = E(H) \setminus \{h'\}$, $E(\widehat{H}_{F'}) = \{h'\}$, and $E(\widehat{H}_F) = \emptyset$ for all other faces F. That is, we now view h' as being associated with F' rather than F^*.

We obtain a feasible flow x for the instance (G, H, u, d) satisfying the nested supports property for this new facial decomposition $\widehat{\mathcal{H}}$ by way of Lemma 17. We define capacities u_F, d_F for each face F as in the statement of Lemma 18 using the new decomposition $\widehat{\mathcal{H}}$. Each instance $(G, \widehat{H}_F, u_F, d_F)$ is feasible. Additionally, each satisfies that $|E(\widehat{H}_F)| < |E(H)|$, and thus $(G, \widehat{H}_F, u_F, d_F)$ has a feasible uncrossed flow y^F by induction. Then the sum of these flows is a feasible uncrossed flow for (G, H, u, d) by Lemma 18. ◀

With a little extra effort, we can modify Lemma 18 and Theorem 19 to show that pairwise-planar instances admit strongly uncrossed flows.

It is straightforward to convert this inductive argument into a polynomial time algorithm, provided we have a polynomial time algorithm to compute a feasible almost-uncrossed flow, as in Lemma 14. However, this is somewhat tricky. To prove Lemma 14, we re-route flow on twice-crossing paths to reduce the total amount of crossing, but the amount of progress depends on the flow values of the crossing paths and could be very small. We defer the details to the full version, but one can efficiently compute an almost-uncrossed flow that is *approximately* feasible - for any $\epsilon > 0$, in polynomial time we can find an almost-uncrossed flow that is feasible for $(G, H, (1+\epsilon)u, d)$. The details are similar to the uncrossing algorithm in [18]. Essentially, by judiciously rounding each path's flow value, we ensure the values are common multiples of a small but sufficiently large fraction to ensure each uncrossing step makes significant progress. We summarize the final result below.

▶ **Theorem 20.** *Let (G, H) be pairwise-planar and let (G, H, u, d) be a congestion instance. Let $\epsilon > 0$ be any constant. If the instance is feasible, then there exists a strongly uncrossed flow f' that is feasible for $(G, H, (1+\epsilon)u, d)$. The flow can be computed in polynomial time.*

Note that $f'/(1+\epsilon)$ is feasible for $(G, H, u, d/(1+\epsilon))$, so this approximate guarantee is useful for maximization as well as congestion. Combining this theorem with our rounding results Theorem 2 and Theorem 3, we obtain Corollary 4.

3.2 Integral Flow-Multicut Gap

In this subsection we prove Corollary 5. Here we are concerned with maximization instances (G, H, u). A *multicut* is a set of supply edges $R \subseteq E(G)$ such that $G - R$ does not contain an st-path for any $(s, t) \in E(H)$. The minimum capacity of a multicut is at least the maximum value of a (fractional) multiflow. The ratio of these quantities is the *(fractional) flow-multicut gap*[12]. The *integral flow-multicut gap* is the ratio of the minimum capacity of a multicut to the maximum value of an integral multiflow.

An immediate corollary of Theorems 19 and 2 is that for pairwise-planar (G, H), the integral flow-multicut gap is bounded by $O(1)$ times the (fractional) flow-multicut gap. We combine this with the following result by Tardos and Vazirani [30] to conclude Corollary 5.

▶ **Theorem 21** (Tardos and Vazirani [30])**.** *For any maximization instance (G, H, u) where G is planar, the (fractional) flow-multicut gap is bounded by $O(1)$.*

[12] This is different from the similarly named *flow-cut gap* (cf. [7]), which is defined for congestion instances and relates the minimum congestion value to sparsity of cuts (not necessarily multicuts).

4 Maximization Setting: Rounding and Hardness

In this section, we prove Theorem 2 and Theorem 6, which cover our rounding and inapproximability results for maximum uncrossed multiflow.

4.1 Uncrossed String Graphs and Approximate Integer Decomposition

First we prove Theorem 2. Our approach is to establish a so-called approximate integer decomposition property [3, 5] for uncrossed multiflows. The outline of the strategy is as follows. Suppose we are given an uncrossed multiflow f, which achieves a value of β. Imagine scaling f by a positive integer k such that kf is integral. Of course, kf may not be feasible. However suppose we are able to show that kf can be decomposed as the sum of a small number b of feasible integral flows. That is $kf = \sum_{i=1}^{b} f^i$ where each f^i is integral and feasible. At least one of the f^i flows achieves a $1/b$ fraction of the value of kf, and hence a k/b fraction of the value of f.

How can we achieve such a decomposition? Suppose (for simplicity) that kf sends 1 unit of flow along paths $\mathcal{P} = \{P_1, \ldots, P_\ell\}$. Furthermore suppose that each edge of G has unit capacity and hence each appears on at most k paths. Our goal then is to partition the paths into classes such that every edge appears at most once in each class; the classes then define the flows f^i. Equivalently, we seek to partition the paths so that paths in the same class do not share an edge. Phrased this way, we can recognize this as a graph colouring problem on the edge-intersection graph of $\mathcal{P}$.

The edge-intersection graph U of $(G, \mathcal{P})$ has node set $\mathcal{P}$, and an edge between distinct $P_1, P_2 \in \mathcal{P}$ if there exists $e \in E(P_1) \cap E(P_2)$. The *load* of $e \in E(G)$ is the number of paths in which e occurs, and the *load* of $(G, \mathcal{P})$ is the maximum load of an edge $e \in E(G)$. Graphs U constructed in this way are a special type of the well-studied family of *string graphs*, which are the intersection graphs of curves in the plane.

▶ **Definition 22** (Uncrossed String Graph, Realization)**.** *An* uncrossed string graph U *is the edge-intersection graph of a family of uncrossed paths* $\mathcal{P}$ *in a planar graph* G*. We say* $(G, \mathcal{P})$ *is a* realization *of* U*.*

Uncrossed string graphs have been studied before for their own sake [11, 32]. Given that U has a load k realization, we want that its chromatic number $\chi(U)$ is some small function of k, ideally $O(k)$. This question is strongly related to the concept of a χ-*bounded* graph family. A family of graphs $\mathcal{G}$ is χ-bounded if there exists a function f such that for all $U \in \mathcal{G}$, $\chi(U) \leq f(\omega(U))$, where $\omega(U)$ is the clique number of U. It is worth noting that general string graphs are not χ-bounded, and it is not obvious that uncrossed string graphs should be. An $O(k)$ bound was conjectured originally in [11], and proven by Fox and Pach [13].

▶ **Theorem 23** (Fox and Pach)**.** *Let* U *be an uncrossed string graph with realization* $(G, \mathcal{P})$ *such that each edge* $e \in E(G)$ *has load at most* k*. Then* $\chi(U) \leq 6ek + 1$*.*

The result of Fox and Pach has not been published, however the proof is elegant and short. It establishes via a probabilistic proof (similar to a standard proof giving a bound on the crossing number of a graph) that U has at most $3ek|\mathcal{P}|$ edges. Hence a greedy colouring procedure that removes the minimum degree vertex (which has degree at most $6ek$) and colours the rest of the graph inductively obtains a $6ek + 1$ colouring. We were not initially aware of Fox and Pach's result and independently proved a weaker bound that $\chi(U) \leq O(k^2 \log |\mathcal{P}|)$. For the sake of completeness we present this proof in the full version.

We now use the colouring result to establish an approximate integer decomposition property and prove a generalization of Theorem 2 with weighted profits on $E(H)$.

▶ **Theorem 24.** *There exists a universal constant $\gamma \geq 1$ such that the following holds. Let (G, H, u) be a planar maximization instance where u is integral, equipped with per-unit profits $w(h) \geq 0$ for $h \in E(H)$. Let f be a feasible uncrossed multiflow. There exists a feasible integral multiflow f^* that is uncrossed and achieves value $|f^*|_w \geq \frac{1}{\gamma}|f|_w$. This solution can be computed in polynomial time.*

Proof. We first present the case where the capacities are unit-valued $u = \vec{1}$. We show how to handle general capacities in the full version - while not hard, it requires some care to handle polynomial time aspects.

Let $\beta = |f|_w$. Let $\mathcal{P}$ be the family of support paths for f. Since $u = \vec{1}$, we have that $f(P) < 1$ for each $P \in \mathcal{P}$. Now, let k be a positive integer such that kf is integral. Since kf is integral, for each P we have that $f(P) = \alpha_P/k$ where $\alpha_P \leq k$ is a positive integer. By treating each P as α_P distinct paths, we may assume that each path routes exactly $\frac{1}{k}$ units of flow. Then, since $\sum_{P:e\in P} f(P) \leq 1$ by feasibility, at most k paths use each edge of G.

Let U be the edge-intersection graph of $(G, \mathcal{P})$, which has load at most k. By Theorem 23, U is b-colourable where $b = O(k)$. Now for each colour class i, we let f_i denote the flow obtained by sending one unit of flow on each path in that colour class, and zero flow on all other paths. Then f_i is integral and satisfies that $\sum_{P:e\in P} f_i(P) \leq 1$ for each $e \in E(G)$, and $kf = \sum_{i=1}^{b} f_i$ since each path's flow was blown up by a factor of k. Since the value of kf is $k\beta$, at least one of the f_i flows has a value of $k\beta/b$. Since $b = O(k)$, then this flow has value $\Omega(1)$ times that of f, as desired.

However, this proof is existential thus far, and does not immediately give a polynomial time algorithm because k could be very large. To fix this, we first have a pre-processing step where we convert f into a feasible flow $\hat{f}$ that uses a subset of f's support paths (so that it remains uncrossed) such that each value $\hat{f}(P)$ is a multiple of $\frac{1}{\text{poly}(n)}$, and so that $|\hat{f}|_w$ is a large fraction of $|f|_w$. We then run the above procedure on $\hat{f}$ instead of f, giving a polynomial time algorithm. We prove the following.

▶ **Lemma 25.** *Let $\epsilon \in (0, 1)$ be any constant. There exists a positive constant N_ϵ such that the following holds. Let $(G, H, u = \vec{1})$ be a unit-capacity planar maximization instance on $n \geq N_\epsilon$ nodes, equipped with per-unit profits $w(h) \geq 0$ for $h \in E(H)$. Let f be a feasible multiflow such that $|f|_w \geq w_{\max} = \max_{h\in E(H)} w(h)$. There exists a multiflow $\hat{f}$ satisfying:*

1. *$\hat{f}$ is feasible,*
2. *the support paths of $\hat{f}$ are a subset of the support paths of f,*
3. *$|\hat{f}|_w \geq (1-\epsilon)|f|_w$, and*
4. *for every path P, $\hat{f}(P)$ is a multiple of $\frac{1}{\log^2 n}$.*

There is a polynomial time algorithm that computes such an $\hat{f}$.

We prove this in the full version using a randomized rounding and re-scaling argument (which can be de-randomized to give a deterministic algorithm). The two conditions that $|f|_w \geq w_{\max}$ and $n \geq N_\epsilon$ are only minor technicalities, as we argue in that version. ◀

It is straightforward to generalize this to the node-capacitated setting - the colouring and integer decomposition property proceed in exactly the same way.

4.2 Inapproximability of Maximum Uncrossed Fractional Multiflow

Next we prove Theorem 6 on the inapproximability of maximum uncrossed (fractional) multiflow. We make use of a strong inapproximability result given by Chuzhoy, Kim, and Nimavat for maximum edge-disjoint paths (MEDP) and maximum node-disjoint paths (MNDP) in planar graphs; i.e. maximum integer multiflow for unit capacities. One minor technicality is that in their formulation, a feasible solution is only allowed to route one unit of flow for each demand. We call these problems *capped MEDP* and *capped MNDP* respectively.

▶ **Theorem 26** (Theorems 1.1 and 1.2, [9]). *For every constant $\epsilon > 0$, there is no $2^{\Omega(\log^{1-\epsilon} n)}$-approximation polynomial-time algorithm for capped MEDP or capped MNDP, assuming that $NP \not\subseteq DTIME(n^{polylog\ n})$. Moreover, there is no $n^{O\left(\frac{1}{(\log\log n)^2}\right)}$-approximation polynomial-time algorithm for capped MEDP or capped MNDP, assuming that for some constant $\delta > 0$, $NP \not\subseteq DTIME(2^{n^\delta})$. These results hold even when the input graph is a maximum degree 3 planar graph, or even if G is a wall graph.*

We now prove Theorem 6 which relies on this result. We focus on the edge-capacitated version (the node-capacitated version is similar).

Proof of Theorem 6. We give a polytime reduction from capped MEDP in wall graphs to maximum uncrossed multiflow in wall graphs, that preserves the approximation ratio up to a constant factor. For an instance (G, H) of the capped MEDP problem where G is a wall graph (which is planar and has maximum degree 3), define the following quantities.

- Let $D_E(G, H)$ denote the maximum number of demands in H which can be routed on edge-disjoint paths in G (i.e. the optimal value for capped MEDP).
- Let $U_E(G, H)$ denote the maximum value of a (fractional) uncrossed multiflow for $(G, H, u = \vec{1})$.

Observe that any capped MEDP solution for (G, H) is a feasible (integral) uncrossed multiflow for $(G, H, u = \vec{1})$. To see this, consider a collection of edge-disjoint paths that form a capped MEDP solution. Recall from Section 2.2 that two edge-disjoint paths cross only if they cross at a node (rather than a longer common subpath). Since G has maximum degree 3, a node crossing is not possible. Thus we have $D_E(G, H) \leq U_E(G, H)$.

Suppose we have a β-approximation algorithm for maximum uncrossed (fractional) multiflow in wall graphs. We obtain an approximation algorithm for capped MEDP as follows. We run our β-approximation algorithm for maximum uncrossed multiflow on $(G, H, u = \vec{1})$ to obtain flow f whose value is at least $U_E(G, H)/\beta$. We apply the polytime algorithm from Theorem 24 to compute a feasible uncrossed integral multiflow f' whose value is $1/\gamma$ that of f. It may still not be a valid capped MEDP solution, since it may route more than one flow path for the same commodity. However, since G has maximum degree 3, it routes on at most 3 paths for each commodity, so by selecting exactly one path for each commodity we obtain a capped MEDP solution whose value is at least $\frac{1}{3}$ that of f'. Thus this algorithm outputs a capped MEDP solution whose value is at least $\frac{U_E(G,H)}{3\gamma\beta} \geq \frac{D_E(G,H)}{3\gamma\beta}$, and is thus a $3\gamma\beta$-approximation algorithm. ◀

5 Congestion Setting: Rounding

In this section, we focus on planar congestion instances (G, H, u, d) where the demands must be fully routed. We show Theorem 3, which states that we can convert (fractional) strongly uncrossed multiflows into integral flows that violate the capacities by a factor of at most 2, and if we accept an additional additive error of less than $d_{\max} = \max_{h \in E(H)} d(h)$ the flow can be made unsplittable. Section 5.1 presents the relevant properties of strongly uncrossed flows, and Section 5.2 gives the proof of Theorem 3.



Figure 7 (Left) The disk expansion of a node z. Paths of the same colour are for the same commodity. The blue and red paths terminate at z, while the green path transits through z. There is no crossing or quasicrossing at z. (Right) We add a node into the disk for each commodity terminating there (blue, red) and connect the new node to the endpoints of that commodity's support paths. Since there is no crossing or quasicrossing at z, this operation can be done without causing edge crossings.

5.1 Clockwise Orderings and Sectors

Let f be a strongly uncrossed multiflow for planar congestion instance (G, H, u, d). Let $\mathcal{P}$ be the set of support paths, and let $\mathcal{P}^h$ be the support paths routing demand $h \in E(H)$. Fix a strongly uncrossed parallelization ϕ. We parallelize the edges and perform disk-expansion at each node (see Section 2.2) so the paths become node-disjoint. Then, for each commodity $h = (s, t) \in E(H)$, we add a node and edges into the disk corresponding to s to join the endpoints of the paths for h, and similarly for the disk of t. This can be done in a planar fashion for all demands simultaneously since the flow is strongly uncrossed (see Section 2.4 and Figure 7). We denote the new graph G_ϕ. For $P \in \mathcal{P}$, we let $\widehat{P}$ be its corresponding path in G_ϕ. For $P, Q \in \mathcal{P}^h$, $\widehat{P}$ and $\widehat{Q}$ are internally node-disjoint paths that share their endpoints. For $P, Q \in \mathcal{P}$ routing different demands, $\widehat{P}$ and $\widehat{Q}$ are (completely) node-disjoint.

Consider a commodity $h = (s, t) \in E(H)$. We use the notation $G_\phi[h]$ to mean the subgraph of G_ϕ induced by the paths $\widehat{\mathcal{P}}^h = \left\{\widehat{P} : P \in \mathcal{P}^h\right\}$. In G_ϕ, the paths of $\widehat{\mathcal{P}}^h$ have a clockwise cyclic ordering about s, say $\widehat{P}_1, \ldots, \widehat{P}_k$. Informally, a *sector* is the region between consecutive paths in the clockwise ordering. We make this more precise as follows.

Consider two paths $\widehat{P}_i$ and $\widehat{P}_j$ in G_ϕ. Orient $\widehat{P}_i$ from s to t, and orient $\widehat{P}_j$ from t to s. The directed curve $\widehat{P}_i \bullet \widehat{P}_j$, obtained by concatenating these oriented curves, is a simple closed (Jordan) curve, and partitions the plane into two regions. We define $\mathrm{CW}^\circ(\widehat{P}_i, \widehat{P}_j)$ to be the set of points q in $\mathbb{R}^2 \setminus (\widehat{P}_i \cup \widehat{P}_j)$ such that q is "to the right" of $\widehat{P}_i \bullet \widehat{P}_j$; that is, if we draw a directed curve from q to $\widehat{P}_i \bullet \widehat{P}_j$ (avoiding $\widehat{P}_i \bullet \widehat{P}_j$ except at the extremity), it hits $\widehat{P}_i \bullet \widehat{P}_j$ from the right. We let $\mathrm{CW}(\widehat{P}_i, \widehat{P}_j)$ denote the closed region $\mathrm{CW}^\circ(\widehat{P}_i, \widehat{P}_j) \cup \widehat{P}_i \cup \widehat{P}_j$. We analogously define $\mathrm{ACW}^\circ(\widehat{P}_i, \widehat{P}_j)$ and $\mathrm{ACW}(\widehat{P}_i, \widehat{P}_j)$ by taking the opposite orientation. Any curve that does not cross and is internally disjoint from $\widehat{P}_i$ and $\widehat{P}_j$ either stays entirely within $\mathrm{CW}(\widehat{P}_i, \widehat{P}_j)$ or stays entirely within $\mathrm{ACW}(\widehat{P}_i, \widehat{P}_j)$.

Now, for our clockwise-ordered family of paths $\widehat{P}_1, \ldots, \widehat{P}_k$, we refer to $\mathrm{CW}^\circ(\widehat{P}_i, \widehat{P}_{i+1})$ for $i = 1, \ldots, k$ (addition modulo k) as the *open sectors* of f^h, and define the *closed sectors* similarly using $\mathrm{CW}(\widehat{P}_i, \widehat{P}_{i+1})$. The open sectors partition $\mathbb{R}^2 \setminus (\widehat{P}_1 \cup \cdots \cup \widehat{P}_k)$. One sector contains the infinite face of $G_\phi[h]$; we call this the *outer sector* of h, and the others the *inner sectors* of h. The *outer paths* of h are the two paths $P, Q \in \mathcal{P}^h$ whose corresponding paths $\widehat{P}, \widehat{Q} \in \widehat{\mathcal{P}}^h$ define the outer sector of h.

Now consider an edge e of the original graph G that appears in $\mathcal{P}^h$. If e lies in any outer path of $\mathcal{P}^h$, it is an *outer edge* of h, otherwise it is an *inner edge* of h. For our congestion result, we exploit the following key properties.

▶ **Lemma 27.** *Let f be a strongly uncrossed multiflow, with a fixed strongly uncrossed parallelization ϕ. For distinct $h, h' \in E(H)$, the sets of inner edges of h and h' are disjoint.*

Proof. To prove the result, we show that if an edge e is shared between the two single-commodity flows in G, then it is an outer edge for one of the commodities.

▷ Claim 28. $G_\phi[h']$ is entirely contained within a single sector of h, and $G_\phi[h]$ is entirely contained within a single sector of h'.

Proof. By symmetry, it suffices to prove the first part. Recall that in G_ϕ, two paths for distinct commodities are completely disjoint, so each individual $\widehat{P} \in \widehat{\mathcal{P}}^{h'}$ lies in at most one sector of h. However, all paths of $\widehat{\mathcal{P}}^{h'}$ share common terminals in G_ϕ, so it must be the same sector for all of $\widehat{\mathcal{P}}^{h'}$. ◁

▷ Claim 29. Either $G_\phi[h']$ is contained in the outer sector of h, or $G_\phi[h]$ is contained in the outer sector of h'.

Proof. By the previous claim and by symmetry, it suffices to show that if $G_\phi[h']$ is contained within an inner sector of h, then $G_\phi[h]$ is contained within the outer sector of h'. Suppose $G_\phi[h']$ is in the inner sector of h defined by paths $\widehat{P}_i$ and $\widehat{P}_{i+1}$. Since $\mathrm{CW}(\widehat{P}_i, \widehat{P}_{i+1})$ is an inner sector, it is bounded, which means $G_\phi[h']$ is contained in the interior of the curve $\widehat{P}_i \bullet \widehat{P}_{i+1}$ rather than the exterior. This implies that $\widehat{P}_i$ and $\widehat{P}_{i+1}$ are contained within the infinite face of $G_\phi[h']$, and so therefore are in the outer sector of h'. Since the entirety of $G_\phi[h]$ is within the same sector of h', it is within the outer sector of h'. ◁

Now, without loss of generality, suppose that $G_\phi[h]$ is contained within the outer sector of h'. Then $G_\phi[h]$ and $G_\phi[h']$ both contain one of the copies of the shared edge e. The copy belonging to h is contained in the infinite face of $G_\phi[h']$, and (by how G_ϕ is constructed) so too is the copy belonging to h'. So then e is an outer edge of h', not an inner edge. ◀

Now, associating the circular ordering $P_1, \ldots, P_k$ to $\mathcal{P}^h$ corresponding to the clockwise ordering $\widehat{P}_1, \ldots, \widehat{P}_k$ of $\widehat{\mathcal{P}}^h$, we prove the second key property.

▶ **Lemma 30.** *Suppose that in G, $P_i, P_j \in \mathcal{P}^h$ share an edge e. One of the following holds:*

- *every path between P_i and P_j in the circular ordering also uses e, or*
- *every path between P_j and P_i in the circular ordering also uses e.*

Proof. In G_ϕ, consider subdividing the copies of e belonging to $\widehat{P}_i$ and $\widehat{P}_j$ by adding one node into each edge. Then add an edge e' between the two new nodes, adding e' such that it only crosses copies of e. The new edge either separates $\mathrm{CW}(\widehat{P}_i, \widehat{P}_j)$ or it separates $\mathrm{ACW}(\widehat{P}_i, \widehat{P}_j)$, where by separates we mean that any path of $\widehat{\mathcal{P}}^h$ that is within the sector must cross e'. The paths between $\widehat{P}_i$ and $\widehat{P}_j$ in the clockwise ordering all lie in $\mathrm{CW}(\widehat{P}_i, \widehat{P}_j)$, and the paths between $\widehat{P}_j$ and $\widehat{P}_i$ in the clockwise ordering all lie in $\mathrm{ACW}(\widehat{P}_i, \widehat{P}_j)$. If a path $\widehat{P}_r$ crosses e', then it means that it uses a copy of e, and hence its corresponding path P_r in G also uses e. The result follows. ◀

5.2 Main Congestion Argument

We are now ready to prove Theorem 3.

Proof of Theorem 3. First, we show how to prove Item 2 (unsplittable flow), since Item 1 ultimately follows from it. Let f be a feasible strongly uncrossed flow with a fixed strongly uncrossed parallelization ϕ. Recall that we let $f(e) = \sum_{P: e \in P} f(P)$ and let f^h be the single-commodity flow for $h \in E(H)$.

Consider a commodity h. Let $P_1, \ldots, P_k$ be the paths of $\mathcal{P}^h$, ordered according to the cyclic clockwise ordering of their corresponding paths $\widehat{P}_1, \ldots, \widehat{P}_k \in \widehat{\mathcal{P}}^h$ in G_ϕ. Re-index so that P_1 and P_k are the outer paths. Let i^* be the smallest index i such that $\sum_{j \le i} f^h(P_j) \ge \frac{d(h)}{2}$. We define $Q^h = P_{i^*}$, and call it the *central path* for h.

▷ Claim 31. If $e \in Q^h$ and e is an outer edge for h (i.e. it lies in P_1 or P_k), then $f^h(e) \ge \frac{d(h)}{2}$.

Proof. Since e is an outer edge, it is in either P_1 or P_k, in addition to $Q^h = P_{i^*}$. From Lemma 30 we have that $e \in P_r$ either for all $1 \le r \le i^*$ or for all $i^* \le r \le k$. In the former case, $f^h(e) \ge \sum_{1 \le r \le i^*} f^h(P_r) \ge \frac{d(h)}{2}$. For the latter case, $f^h(e) \ge \sum_{i^* \le r \le k} f^h(P_r) = d(h) - \sum_{1 \le r < i^*} f^h(P_r) > d(h) - \frac{d(h)}{2} = \frac{d(h)}{2}$, where the last inequality follows since $\sum_{1 \le r < i^*} f^h(P_r) < \frac{d(h)}{2}$ by definition of i^*. ◁

To obtain an unsplittable flow f', for each $h \in E(H)$ we route $d(h)$ units of flow along the path Q^h. For the congestion bound, consider an arbitrary edge $e \in E(G)$. We will bound the amount of flow through e. Let $H_e = \{h \in E(H) : e \in Q^h\}$, so $f'(e) = \sum_{h \in H_e} d(h)$. Define $O_e = \{h \in H_e : e \text{ is an outer edge of } h\}$ and also $I_e = \{h \in H_e : e \text{ is an inner edge of } h\}$. These sets partition H_e, so $f'(e) = \sum_{h \in O_e} d(h) + \sum_{h \in I_e} d(h)$. We know that $|I_e| \le 1$ from Lemma 27. Additionally, by Claim 31, for each $h \in O_e$ we have that $f^h(e) \ge \frac{d(h)}{2}$. We now split into two cases.

Case 1: $\sum_{h \in O_e} \frac{d(h)}{2} < u(e)$. Then,

$$f'(e) < 2u(e) + \sum_{h \in I_e} d(h) \le 2u(e) + d_{\max}.$$

Case 2: $\sum_{h \in O_e} \frac{d(h)}{2} \ge u(e)$. Then $\sum_{h \in O_e} f^h(e) \ge \sum_{h \in O_e} \frac{d(h)}{2} \ge u(e)$, so the commodities of O_e saturate e in f. It follows that $I_e = \emptyset$. Hence,

$$f'(e) = \sum_{h \in O_e} d(h) \le \sum_{h \in O_e} 2f^h(e) \le 2f(e) \le 2u(e) < 2u(e) + d_{\max}.$$

This concludes the proof of Item 2. The above is easily turned into a polynomial time algorithm. We show how to prove Item 1 from Item 2 in the full version. ◀

References

1. Cédric Bentz. Disjoint paths in sparse graphs. *Discrete Applied Mathematics*, 157(17):3558–3568, October 2009. doi:10.1016/j.dam.2009.03.009.
2. Amit Chakrabarti, Lisa Fleischer, and Christophe Weibel. When the cut condition is enough: A complete characterization for multiflow problems in series-parallel networks. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 19–26. ACM, 2012. doi:10.1145/2213977.2213980.
3. Chandra Chekuri, Marcelo Mydlarz, and F Bruce Shepherd. Multicommodity demand flow in a tree and packing integer programs. *ACM Transactions on Algorithms (TALG)*, 3(3):27, 2007. doi:10.1145/1273340.1273343.
4. Chandra Chekuri, Guyslain Naves, and F Bruce Shepherd. Maximum edge-disjoint paths in k-sums of graphs. In *International Colloquium on Automata, Languages, and Programming*, pages 328–339. Springer, 2013. doi:10.1007/978-3-642-39206-1_28.
5. Chandra Chekuri and F Bruce Shepherd. Approximate integer decompositions for undirected network design problems. *SIAM Journal on Discrete Mathematics*, 23(1):163–177, 2009. doi:10.1137/040617339.

6 Chandra Chekuri, F Bruce Shepherd, and Christophe Weibel. Flow-cut gaps for integer and fractional multiflows. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1198–1208. Society for Industrial and Applied Mathematics, 2010. doi:10.1137/1.9781611973075.96.

7 Chandra Chekuri, F Bruce Shepherd, and Christophe Weibel. Flow-cut gaps for integer and fractional multiflows. *Journal of Combinatorial Theory, Series B*, 103(2):248–273, 2013. doi:10.1016/j.jctb.2012.11.002.

8 Julia Chuzhoy, David HK Kim, and Rachit Nimavat. Almost polynomial hardness of node-disjoint paths in grids. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1220–1233, 2018.

9 Julia Chuzhoy, David Hong Kyun Kim, and Rachit Nimavat. Almost polynomial hardness of node-disjoint paths in grids. *Theory of Computing*, 17(6):1–57, 2021. doi:10.4086/toc.2021.v017a006.

10 Denis Cornaz. Max-multiflow/min-multicut for G + H series-parallel. *Discrete Mathematics*, 311(17):1957–1967, September 2011. doi:10.1016/j.disc.2011.05.025.

11 Louis Esperet, Daniel Gonçalves, and Arnaud Labourel. Coloring a set of touching strings. *Electronic Notes in Discrete Mathematics*, 34:213–217, August 2009. doi:10.1016/j.endm.2009.07.035.

12 David Alemán Espinosa, Nikhil Kumar, Joseph Poremba, and Bruce Shepherd. Unsplittable flow cut gap in undirected graphs. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1570–1605. SIAM, 2026. doi:10.1137/1.9781611978971.57.

13 Jacob Fox and János Pach. Touching Strings. Unpublished manuscript, 2012.

14 N. Garg, V. V. Vazirani, and M. Yannakakis. Primal-dual approximation algorithms for integral flow and multicut in trees. *Algorithmica*, 18(1):3–20, May 1997. doi:10.1007/BF02523685.

15 Naveen Garg, Nikhil Kumar, and András Sebő. Integer plane multiflow maximisation: one-quarter-approximation and gaps. *Mathematical Programming*, 195(1):403–419, 2022. doi:10.1007/S10107-021-01700-8.

16 Michel X Goemans and David P Williamson. Primal-dual approximation algorithms for feedback problems in planar graphs. *Combinatorica*, 1(18):37–59, 1998. doi:10.1007/PL00009810.

17 Chien-Chung Huang, Mathieu Mari, Claire Mathieu, Kevin Schewior, and Jens Vygen. An approximation algorithm for fully planar edge-disjoint paths. *SIAM Journal on Discrete Mathematics*, 35(2):752–769, 2021. doi:10.1137/20M1319401.

18 Chien-Chung Huang, Mathieu Mari, Claire Mathieu, and Jens Vygen. Approximating maximum integral multiflows on bounded genus graphs. *Discrete & Computational Geometry*, 70(4):1266–1291, 2023. doi:10.1007/S00454-023-00552-7.

19 Philip Klein, Serge A. Plotkin, and Satish Rao. Excluded minors, network decomposition, and multicommodity flow. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, STOC '93, pages 682–690, New York, NY, USA, 1993. ACM. doi:10.1145/167088.167261.

20 Bernhard Korte, Làszlo Lovàsz, Hans Jürgen Prömel, and Alexander Schrijver. *Paths, flows, and VLSI-layout*. Springer-Verlag, 1990.

21 Kazuhiko Matsumoto, Takao Nishizeki, and Nobuji Saito. Planar multicommodity flows, maximum matchings and negative cycles. *SIAM Journal on Computing*, 15(2):495–510, 1986. doi:10.1137/0215034.

22 Guyslain Naves. The hardness of routing two pairs on one face. *Mathematical Programming*, 131(1-2):49–69, February 2012. doi:10.1007/s10107-010-0343-0.

23 Janos Pách. personal communication.

24 Niklas Schlomberg. An improved integrality gap for disjoint cycles in planar graphs. *arXiv preprint*, 2024. arXiv:2404.17813.

25 Niklas Schlomberg, Hanjo Thiele, and Jens Vygen. Packing cycles in planar and bounded-genus graphs. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2069–2086. SIAM, 2023. doi:10.1137/1.9781611977554.CH79.

26 Alexander Schrijver. Homotopic routing methods. *Paths, Flows, and VLSI-layout*, pages 329–371, 1990.

27 Alexander Schrijver. Disjoint homotopic paths and trees in a planar graph. *Discrete & Computational Geometry*, 6:527–574, 1991. doi:10.1007/BF02574704.

28 Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer, 2003.

29 Paul D Seymour. Matroids and multicommodity flows. *European Journal of Combinatorics*, 2(3):257–290, 1981. doi:10.1016/S0195-6698(81)80033-9.

30 Éva Tardos and Vijay V. Vazirani. Improved bounds for the max-flow min-multicut ratio for planar and K-free graphs. *Information Processing Letters*, 47(2):77–80, August 1993. doi:10.1016/0020-0190(93)90228-2.

31 Vera Traub, Laura Vargas Koch, and Rico Zenklusen. Single-source unsplittable flows in planar graphs. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 639–668. SIAM, 2024. doi:10.1137/1.9781611977912.24.

32 Wouter Cames Van Batenburg, Louis Esperet, and Tobias Müller. Coloring Jordan Regions and Curves. *SIAM Journal on Discrete Mathematics*, 31(3):1670–1684, January 2017. doi:10.1137/16M1092726.

33 Jens Vygen. Np-completeness of some edge-disjoint paths problems. *Discrete Applied Mathematics*, 61(1):83–90, 1995. doi:10.1016/0166-218X(93)E0177-Z.

Approximating (Weighted) Chromatic Correlation Clustering via Cluster LP

Fateme Abbasi[1]
Institute of Computer Science, University of Wrocław, Poland

Hyung-Chan An[1]
Department of Computer Science and Engineering, Yonsei University, Seoul, South Korea

Jarosław Byrka[1]
Institute of Computer Science, University of Wrocław, Poland

Changyeol Lee[1]
Department of Computer Science and Engineering, Yonsei University, Seoul, South Korea

Yongho Shin[1]
Institute of Computer Science, University of Wrocław, Poland

Abstract

Correlation Clustering is a fundamental clustering problem that is generalized to Chromatic Correlation Clustering to incorporate categorical data. Both problems have been intensively studied, and recently, substantial improvements were obtained in the approximation algorithms for Correlation Clustering. At the heart of this success lies a new linear program (LP) formulation called the *cluster LP*; a natural question was whether this LP can be extended to Chromatic Correlation Clustering to enable similar success.

We answer this question in the affirmative by presenting a $(2+\varepsilon)$-approximation algorithm for the problem using a *chromatic* cluster LP. We then consider Weighted Chromatic Correlation Clustering, in which edges have fractional weights satisfying the probability constraints, to show that our algorithm extends to this weighted version to yield the same approximation guarantee.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Chromatic correlation clustering, Chromatic cluster LP, Preclustering

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.13

Category APPROX

Related Version *Full Version*: `https://arxiv.org/abs/2510.13446` [1]

Funding Supported by NCN grant number 2020/39/B/ST6/01641. This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2021-II212068, Artificial Intelligence Innovation Hub). This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2025-00563707). This work was partly supported by an IITP grant funded by the Korean Government (MSIT) (No. RS-2020-II201361, Artificial Intelligence Graduate School Program (Yonsei University)).

Acknowledgements The authors thank the anonymous reviewers for their helpful comments. Part of this work was done while H.-C. An and C. Lee were visiting the University of Wrocław, Poland.

[1] Equal contribution

© Fateme Abbasi, Hyung-Chan An, Jarosław Byrka, Changyeol Lee, and Yongho Shin;
licensed under Creative Commons License CC-BY 4.0

1 Introduction

Given a complete graph where each edge is labeled by either "+" or "−", Correlation Clustering (CC) asks to find a vertex clustering that minimizes the total number of *disagreed edges*, where we say an edge is disagreed if it is a "−" edge within a cluster or a "+" edge across clusters. This problem is a fundamental combinatorial optimization problem that has been extensively studied in various forms across fields [28, 23, 15, 7, 12, 24, 6]. The readers are also referred to Wahid and Hassini [26] for a dedicated survey on this problem.

This problem is well-known to be NP/APX-hard [24, 6, 12, 10], and recently, it is further shown that the problem cannot be approximated within a constant smaller than 24/23 unless P = BPP [9]. There has also been a long line of work [6, 10, 2, 11, 14, 13, 9] on improving the approximation ratio for this problem that recently culminated in 1.485 due to Cao, Cohen-Addad, Lee, Li, Newman, and Vogl [9]. A key component of their positive result is in approximately solving an exponential-sized LP, called the *cluster LP*, in polynomial time and rounding this nearly optimal solution to an integral solution.

Chromatic Correlation Clustering (CCC) is a generalization of CC motivated by clustering data with categorical interactions [8, 5, 3]. In this generalization, "+" edges are labeled with colors. The objective is to find a clustering of vertices and a coloring of the clusters that minimize the total number of disagreements, where "+" edges within a cluster are deemed disagreed if their colors differ from that of the cluster. This chromatic version has been intensively studied in the literature [8, 4, 18, 27, 20], but previous approaches to this generalization were nevertheless limited to color-blind algorithms [18], reduction to the monochromatic version [4, 18, 27], and rounding algorithms using the standard LP relaxation [4]. The recent work by Lee, Fan, and Lee [20] achieving an approximation ratio of 2.15 for CCC is also based on reducing to monochromatic CC using an optimal solution to the standard LP relaxation for CCC.

In light of success of the cluster LP in CC, it is interesting to see if a chromatic version of the cluster LP helps improve the approximation algorithm for CCC. In this paper, we affirmatively answer this question: we show that a natural adaptation of the cluster LP, called the *chromatic cluster LP*, can be solved nearly optimally in polynomial time, and any feasible solution to the chromatic cluster LP can be rounded into an integral solution of cost within a factor of 2. These together yield a $(2+\varepsilon)$-approximation algorithm for CCC, improving upon 2.15 due to Lee et al. [20]. Furthermore, we identify how we can generalize this approach to the weighted case of CCC.

In addition to CCC, Weighted (Monochromatic) Correlation Clustering [6, 17, 2, 25, 11, 21] is another well-studied generalization of CC. In this problem, rather than "integrally" declaring an edge as "+" or "−", each edge is now assigned a fractional "+" weight and "−" weight. These two weights must be nonnegative and sum to 1. This condition is thereby called the *probability constraint*. The objective of the problem is to find a clustering of vertices that minimizes the total *weight* of disagreements. It is worth noting that there exists an approximation-preserving reduction to unweighted CC due to Chawla, Makarychev, Schramm, and Yaroslavtsev [11]. This reduction is based on replacing each vertex in the original weighted instance by a sufficiently large clique of "+" edges while the numbers of "+" and "−" edges between two cliques are proportional to the weights between the corresponding vertices.

In this paper, we study the weighted generalization of CCC, called Weighted Chromatic Correlation Clustering (WCCC), which is naturally defined as follows: each edge is assigned "+" weights, one for each color, and a (colorless) "−" weight. These weights

must respect the probability constraint. The goal is to find a clustering that minimizes the total weighted disagreements. We demonstrate that our approach for CCC smoothly extends to this weighted generalization, yielding an approximation algorithm with the same factor of $(2+\varepsilon)$. The aforementioned approximation-preserving reduction for the monochromatic version [11] unfortunately does not easily extend to this chromatic version because it is unclear how one could assign colors to the edges within each clique originated from a vertex in the original instance.

Recently, Fan, Lee, and Lee [16] presented a 1.64-approximation algorithm for unweighted CCC. They also approximately solve the same chromatic cluster LP in polynomial time using a slightly different approach from ours, and then round this solution with both a cluster-based algorithm and a pivot-based algorithm, yielding the 1.64-approximation algorithm. One notable difference in solving the chromatic cluster LP lies in the construction of *admissible edges* (see Section 5.2): they adapted Cao et al.'s construction [9] into a color-sensitive version with parallel admissible edges introduced, while we show that the original construction is sufficient. Furthermore, we extend our results to the weighted setting, demonstrating the applicability of this approach to other settings. We note that our results for CCC presented in this paper were obtained independently, whereas our results for WCCC were obtained after the arXiv announcement [19] of the unweighted results of Fan et al [16].

2 Problem definitions

We begin with defining Chromatic Correlation Clustering (CCC). An instance of CCC consists of a set L of colors and a complete graph (V, E) where each edge is specified as either a $+$ edge or a $-$ edge, and each $+$ edge e is associated with a color $c_e \in L$. Let E^ℓ be the set of edges colored ℓ; let $E^+ := \bigcup_{\ell \in L} E^\ell$ and $E^- := E \setminus E^+$ denote the sets of $+$ and $-$ edges, respectively. Given an instance $(V, E, L, \{c_e\}_{e \in E^+})$ of the problem, a (feasible) solution consists of a partition $\mathcal{C}$ of V and a coloring function $\chi : \mathcal{C} \to L$ that assigns a color to each part. Given a solution $(\mathcal{C}, \chi)$, an edge $e = uv$ is called a *disagreement* if and only if one of the following conditions holds:

- e is a $-$ edge and both u and v belong to the same cluster $C \in \mathcal{C}$;
- e is a $+$ edge and u and v belong to different clusters $C \neq C' \in \mathcal{C}$; and
- e is a $+$ edge and both u and v belong to the same cluster $C \in \mathcal{C}$, but $\chi(C) \neq c_e$.

We also refer to a disagreement simply as a *disagreed* edge. If an edge is not a disagreement, we call it an *agreement* or an *agreed* edge. The goal of the problem is to find a solution that minimizes the total number of disagreements, which is also referred to as the cost of the solution.

We now turn our attention to Weighted Chromatic Correlation Clustering (WCCC). An instance of WCCC consists of a complete graph where each edge e is associated with $|L \cup \{-\}|$ *weights*, denoted by $\{w^\ell(e)\}_{\ell \in L \cup \{-\}}$. The weights of each edge satisfy the *probability constraint*, i.e., $w^\ell(e) \geq 0$ for every $\ell \in L \cup \{-\}$ and $\sum_{\ell \in L \cup \{-\}} w^\ell(e) = 1$. A (feasible) solution to a WCCC instance (V, E, L, w) also consists of a partition $\mathcal{C}$ of V and a coloring function $\chi : \mathcal{C} \to L$. To define the total cost of a solution $(\mathcal{C}, \chi)$, we define the cost incurred by an edge $e = uv$ in the solution, denoted by $\mathsf{cost}_{(\mathcal{C},\chi)}(e)$, as follows:

- $\mathsf{cost}_{(\mathcal{C},\chi)}(e) := \sum_{\ell \in L} w^\ell(e) = 1 - w^-(e)$ if u and v belong to different clusters in $\mathcal{C}$;
- $\mathsf{cost}_{(\mathcal{C},\chi)}(e) := \sum_{\ell \in L \cup \{-\} \setminus \{\chi(C)\}} w^\ell(e) = 1 - w^{\chi(C)}(e)$ if both u and v belong to a cluster $C \in \mathcal{C}$.

The cost of a solution $(\mathcal{C}, \chi)$ is then defined to be the sum of costs incurred by all edges, i.e., $\sum_{e \in E} \mathsf{cost}_{(\mathcal{C},\chi)}(e)$. We may omit the subscript $(\mathcal{C}, \chi)$ when it is clear from context. The objective of the problem is to find a solution of the minimum cost.

We further define the notation that will be used. Let $n := |V|$ denote the number of vertices. For the unweighted CCC problem, let $E^{-\ell} := E \setminus E^{\ell}$ for every $\ell \in L$. Given $F \subseteq E$, $u \in V$, and $S \subseteq V$, let $F(u,S) := \{uv \in F \mid v \in S\}$; for example, given $u \in V$ and $S \subseteq V$, $E^{+}(u,S)$ denotes the set of $+$ edges whose one endpoint is u and the other endpoint is in S.

On the other hand, for WCCC, given $U \subseteq L \cup \{-\}$ and $S,T \subseteq V$, let $w^{U}(S,T) := \sum_{\ell \in U} \sum_{uv:u \in S, v \in T} w^{\ell}(uv)$. For notational simplicity, let $w^{U}(u,T)$ and $w^{U}(u,v)$, respectively, denote $w^{U}(\{u\},T)$ and $w^{U}(\{u\},\{v\})$. For any edge $e = uv$, $w^{U}(u,v)$ can be written as $w^{U}(e)$. Note that $w^{U}(u,u) = 0$ by definition. For $\ell \in L$, let $w^{-\ell}(S,T) := w^{L \cup \{-\} \setminus \{\ell\}}(S,T)$. Similarly, let $w^{+}(S,T) := w^{L}(S,T)$.

3 A constant-factor approximation algorithm for WCCC

This section is devoted to a constant-factor approximation algorithm for WCCC. The algorithm is a reduction to CCC inspired by Bansal, Blum, and Chawla [6]. Recall that constant-factor approximation algorithms [4, 18, 27, 20] are known for CCC.

Given an instance $I_{\mathsf{ori}} = (V, E, L, w_{\mathsf{ori}})$ of WCCC, we construct a new instance $I_{\mathsf{new}} = (V, E^{+} \cup E^{-}, L, c)$ for CCC as follows:

$$E^{-} := \{e \in E \mid \arg\max_{\ell \in L \cup \{-\}} \{w^{\ell}_{\mathsf{ori}}(e)\} = -\} \text{ and}$$
$$c_e := \arg\max_{\ell \in L} \{w^{\ell}_{\mathsf{ori}}(e)\} \text{ for } e \in E^{+} := E \setminus E^{-},$$

where ties are broken arbitrarily. We then run a ρ-approximation algorithm for CCC on I_{new}, and return the output of this algorithm.

Let $\mathsf{cost}_{\mathsf{ori}}(\cdot)$ and $\mathsf{cost}_{\mathsf{new}}(\cdot)$ denote the total cost of a solution with respect to I_{ori} and I_{new}, respectively. Let $\mathrm{OPT}_{\mathsf{ori}}$ and $\mathrm{OPT}_{\mathsf{new}}$ denote optimal solutions to I_{ori} and I_{new}, respectively, and let SOL denote the final output of the algorithm. Note that $\mathsf{cost}_{\mathsf{new}}(\mathrm{SOL}) \le \rho\, \mathsf{cost}_{\mathsf{new}}(\mathrm{OPT}_{\mathsf{new}})$ by the execution.

▶ **Lemma 1.** $\mathsf{cost}_{\mathsf{ori}}(\mathrm{SOL}) \le \mathsf{cost}_{\mathsf{new}}(\mathrm{SOL}) + \mathsf{cost}_{\mathsf{ori}}(\mathrm{OPT}_{\mathsf{ori}})$.

Proof. Fix any edge $e \in E$. If e is disagreed by SOL in I_{new}, it incurs a unit cost in I_{new}. Since the cost incurred by any edge in I_{ori} is trivially bounded by 1, the total cost incurred by the edges disagreed by SOL is at most $\mathsf{cost}_{\mathsf{new}}(\mathrm{SOL})$.

On the other hand, if e is agreed by SOL in I_{new}, we cannot use the above argument because it incurs no cost in I_{new}. Observe however that this edge incurs $1 - w^{\ell}_{\mathsf{ori}}(e)$ in SOL in I_{ori}, where $\ell = \arg\max_{\ell' \in L \cup \{-\}} \{w^{\ell'}_{\mathsf{ori}}(e)\}$ by the construction of I_{new}. Note that, in I_{ori}, any solution must incur $1 - w^{\ell}_{\mathsf{ori}}(e)$ from e due to the choice of ℓ. Hence, the total cost incurred by the edges agreed by SOL is bounded from above by $\mathsf{cost}_{\mathsf{ori}}(\mathrm{OPT}_{\mathsf{ori}})$. ◀

▶ **Lemma 2.** *For any solution* $\mathcal{S}$, $\mathsf{cost}_{\mathsf{new}}(\mathcal{S}) \le 2\, \mathsf{cost}_{\mathsf{ori}}(\mathcal{S})$.

Proof. Fix any edge $e \in E$ disagreed by $\mathcal{S}$ in I_{new}. Let $\ell := -$ if $e \in E^{-}$ and $\ell := c_e$ if $e \in E^{+}$. Observe that $w^{\ell'}_{\mathsf{ori}}(e) \le \frac{1}{2}$ for any $\ell' \in L \cup \{-\} \setminus \{\ell\}$ due to the construction of I_{new}. Hence, the cost incurred by e in $\mathcal{S}$ in I_{ori} is then at least $\min_{\ell' \in L \cup \{-\} \setminus \{\ell\}} \{1 - w^{\ell'}_{\mathsf{ori}}(e)\} \ge \frac{1}{2}$. ◀

▶ **Theorem 3.** *This algorithm is a* $(2\rho+1)$*-approximation algorithm for WCCC.*

Proof. We can derive that

$$\begin{aligned}
\mathsf{cost}_{\mathsf{ori}}(\mathrm{SOL}) &\le \mathsf{cost}_{\mathsf{new}}(\mathrm{SOL}) + \mathsf{cost}_{\mathsf{ori}}(\mathrm{OPT}_{\mathsf{ori}}) \\
&\le \rho\, \mathsf{cost}_{\mathsf{new}}(\mathrm{OPT}_{\mathsf{new}}) + \mathsf{cost}_{\mathsf{ori}}(\mathrm{OPT}_{\mathsf{ori}}) \\
&\le \rho\, \mathsf{cost}_{\mathsf{new}}(\mathrm{OPT}_{\mathsf{ori}}) + \mathsf{cost}_{\mathsf{ori}}(\mathrm{OPT}_{\mathsf{ori}}) \\
&\le (2\rho+1)\, \mathsf{cost}_{\mathsf{ori}}(\mathrm{OPT}_{\mathsf{ori}}),
\end{aligned}$$

where the first inequality is due to Lemma 1, and the last is due to Lemma 2. ◀

4 Chromatic cluster LPs

In this section, we define the chromatic cluster LPs for CCC and WCCC, respectively, and present a rounding algorithm whose expected cost is at most twice the cost of the given LP solution.

Chromatic cluster LPs. Define the *chromatic cluster LP* for CCC as follows:

$$\begin{aligned} \min\ & \textstyle\sum_{S\subseteq V,\ell\in L}\left(\frac{|\delta^+(S)|}{2}+|E^{-\ell}[S]|\right)z_S^\ell \\ \text{s.t.}\ & \textstyle\sum_{S\ni v,\ell\in L} z_S^\ell = 1, && \forall v\in V, \\ & z_S^\ell \geq 0, && \forall S\subseteq V\,\forall \ell\in L, \end{aligned}$$

where $z_S^\ell = 1$ indicates whether S is a cluster of color ℓ, $\delta^+(S) := \{(u,v)\in E^+ \mid u\in S, v\notin S\}$, and $E^{-\ell}[S] := \binom{S}{2}\cap E^{-\ell}$. Let $t_v^\ell := 1-\sum_{S\ni v} z_S^\ell$ indicate whether v is *not* colored ℓ and $x_{uv}^\ell := 1-\sum_{S\supseteq uv} z_S^\ell$ denote whether u and v are *not* together in a same cluster colored ℓ. Observe that

$$\textstyle\sum_{S\ni u,S\not\ni v} z_S^\ell = \sum_{S\ni u} z_S^\ell - \sum_{S\supseteq uv} z_S^\ell = x_{uv}^\ell - t_u^\ell, \qquad \forall \ell\in L\,\forall uv\in E \tag{1}$$

$$\textstyle\sum_{\ell\in L} t_v^\ell = |L| - \sum_{\ell\in L, S\ni v} z_S^\ell = |L|-1, \qquad \forall v\in V. \tag{2}$$

Using these equalities, we can rephrase the objective function as $\mathrm{obj}(x) := \sum_{uv\in E^+} x_{uv}^{c_{uv}} + \sum_{uv\in E^-}\sum_{\ell\in L}(1-x_{uv}^\ell)$. We defer its derivation to Appendix A.

The chromatic cluster LP for WCCC has the same sets of variables and constraints, while the objective function is replaced with

$$\textstyle\sum_{S\subseteq V,\ell\in L}\left(\frac{w^+(S,V\setminus S)}{2}+w^{-\ell}(S,S)\right)z_S^\ell.$$

Through a similar derivation to the unweighted version, the objective function can be rephrased as $\mathrm{obj}(x) := \sum_{uv\in E,\ell\in L}\left(w^\ell(uv)x_{uv}^\ell + w^-(uv)(1-x_{uv}^\ell)\right)$. The derivation can also be found in Appendix A.

Cluster-based algorithm. The algorithm is the same for CCC and WCCC. Given a solution z feasible to the chromatic cluster LP, consider the following algorithm. Let V' be the currently unclustered vertices (initially, $V' := V$). Until V' becomes empty, we randomly sample a cluster $S\subseteq V$ colored ℓ with probability $\frac{z_S^\ell}{\sum_{\ell'\in L, S'\subseteq V} z_{S'}^{\ell'}}$. We then cluster $V'\cap S$ colored ℓ and update $V' := V'\setminus S$ accordingly. This ends the description of the cluster-based algorithm.

We begin by analyzing the algorithm for CCC. Recall that $x_{uv}^\ell = 1-\sum_{S\supseteq uv} z_S^\ell$ for every $\ell\in L$ and $uv\in E$. For notational simplicity, let $x_{uv} := 1-\sum_{\ell\in L, S\supseteq uv} z_S^\ell$ indicate whether u and v are *not* clustered together (regardless of its color). Note that we have $1-x_{uv} = \sum_{\ell\in L}(1-x_{uv}^\ell)$, implying that, for any $\ell\in L$, $x_{uv}\leq x_{uv}^\ell$. The following lemma is useful in showing the approximation ratios, whose proof is deferred to Appendix A.

▶ **Lemma 4.** *For every $uv\in E$ and color $\ell\in L$, the probability that uv is in a same cluster colored ℓ is $\frac{1-x_{uv}^\ell}{1+x_{uv}}$, and the probability that uv is separated is $\frac{2x_{uv}}{1+x_{uv}}$.*

▶ **Theorem 5.** *Given a feasible solution z to the chromatic cluster LP for CCC, the cluster-based algorithm outputs an integral solution $(\mathcal{C},\chi)$ whose expected cost is at most twice the objective value of z. The algorithm terminates in time $\mathsf{poly}(n,|\mathsf{supp}(z)|)$ with high probability, where $\mathsf{supp}(z)$ denotes the support set of z.*

Proof. From Lemma 4, we can derive that the probability that $uv \in E$ is not in a cluster colored ℓ is at most $1 - \frac{1-x^\ell_{uv}}{1+x_{uv}} \leq \frac{2x^\ell_{uv}}{1+x_{uv}}$. Hence, the expected cost of $(\mathcal{C}, \chi)$ for CCC is bounded by

$$\begin{aligned}
&\textstyle\sum_{uv\in E^+} \Pr[uv \text{ is not in a same cluster } C \text{ of color } \chi(C) = c_{uv}] \\
&\qquad + \textstyle\sum_{uv\in E^-} \Pr[uv \text{ is not separated}] \\
&\leq \textstyle\sum_{uv\in E^+} \frac{2x^{c_{uv}}_{uv}}{1+x_{uv}} + \sum_{uv\in E^-} \frac{1-x_{uv}}{1+x_{uv}} \\
&\leq 2\textstyle\sum_{uv\in E^+} x^{c_{uv}}_{uv} + \sum_{uv\in E^-} (1 - x_{uv}) \\
&\leq 2\textstyle\sum_{uv\in E^+} x^{c_{uv}}_{uv} + \sum_{uv\in E^-} \sum_{\ell\in L}(1 - x^\ell_{uv}) \leq 2\,\mathrm{obj}(x).
\end{aligned}$$

To show that the algorithm terminates in time $\mathsf{poly}(n, |\mathsf{supp}(z)|)$, observe that, for any vertex $v \in V$, the probability of v being clustered in each iteration is $\frac{\sum_{S\ni v,\ell\in L} z^\ell_S}{\sum_{\ell\in L, S\subseteq V} z^\ell_S} \geq \frac{1}{|\mathsf{supp}(z)|}$. Hence, using the union bound, we can show the algorithm terminates in $\Theta(|\mathsf{supp}(z)| \log n)$ iterations with high probability. ◀

For WCCC, note that Lemma 4 is still satisfied because the algorithm is equivalent. We thus have the weighted version for Theorem 5 as below.

▶ **Theorem 6.** *Given a feasible solution z to the chromatic cluster LP for WCCC, the cluster-based algorithm outputs an integral solution $(\mathcal{C}, \chi)$ whose expected cost is at most twice of the objective value of z. The algorithm terminates in time $\mathsf{poly}(n, |\mathsf{supp}(z)|)$ with high probability.*

Proof. For WCCC, observe that the objective value can be written as

$$\begin{aligned}
\mathrm{obj}(x) &= \textstyle\sum_{e\in E,\ell\in L} \left(w^\ell(e)x^\ell_e + w^-(e)(1 - x^\ell_e)\right) \\
&= \textstyle\sum_{e\in E} \left[\sum_{\ell\in L} w^\ell(e)x^\ell_e + w^-(e)(1 - x_e)\right].
\end{aligned}$$

It thus suffices to show that the expected cost incurred by each $e \in E$ is bounded by twice $\sum_{\ell\in L} w^\ell(e)x^\ell_e + w^-(e)(1 - x_e)$. Observe that

$$\begin{aligned}
\mathbb{E}[\mathsf{cost}(e)] &= \textstyle\sum_{\ell\in L} \left[w^\ell(e)\Pr[e \text{ separated}] + (1 - w^\ell(e))\Pr[e \subseteq C \wedge \chi(C) = \ell]\right] \\
&\leq \textstyle\sum_{\ell\in L} \left[w^\ell(e)\,2x_e + (1 - w^\ell(e))\,(1 - x^\ell_e)\right] \\
&= \textstyle\sum_{\ell\in L} \left[w^\ell(e)(x_e + x^\ell_e) + (1 - x^\ell_e) - w^\ell(e)(1 - x_e)\right] \\
&= \textstyle\sum_{\ell\in L} w^\ell(e)(x_e + x^\ell_e) + w^-(e)(1 - x_e) \\
&\leq 2\textstyle\sum_{\ell\in L} w^\ell(e)x^\ell_e + w^-(e)(1 - x_e),
\end{aligned}$$

where the first inequality follows from Lemma 4, the last equality from the fact that $w^-(e) = 1 - \sum_{\ell\in L} w^\ell(e)$ and $1 - x_e = \sum_{\ell\in L}(1 - x^\ell_e)$, and the last inequality from the fact that $x_e \leq x^\ell_e$ for any $\ell \in L$.

The same argument used in the proof of Theorem 5 for termination completes the proof. ◀

5 Preclustering

Due to Theorems 5 and 6, to obtain a $(2 + \varepsilon)$-approximation algorithm, it suffices to argue that a nearly optimal solution to the chromatic cluster LPs can be found in polynomial time. To this end, we adapt the notion of a *preclustered instance* [13, 9] to the chromatic versions as follows.

▶ **Definition 7** (Preclustered instance). *Given an instance, a* preclustered instance *is defined by a triple* $(\mathcal{K}, \chi_{\mathsf{pre}}, E_{\mathsf{adm}})$ *such that*

- $\mathcal{K}$ *is a partition of* V*;*
- $\chi_{\mathsf{pre}} : \mathcal{K}_{>1} \to L$ *is a color assignment for every non-singleton part in* $\mathcal{K}$*, where we denote by* $\mathcal{K}_{>1} := \{K \in \mathcal{K} \mid |K| > 1\}$ *the set of non-singleton parts of* $\mathcal{K}$*; and*
- $E_{\mathsf{adm}} \subseteq \{uv \in \binom{V}{2} \mid \nexists K \in \mathcal{K} \text{ such that } uv \subseteq K\}$ *is a set of edges across* $\mathcal{K}$*.*

We call $\mathcal{K}$ *and* E_{adm} *a* preclustering *and* admissible *edges, respectively. Each part in* $\mathcal{K}$ *is called a* precluster*. Any pair* uv *of two vertices* u *and* v *in a same precluster is called an* atomic *edge. Any pair that is neither atomic nor admissible is called a* non-admissible *edge.*

▶ **Definition 8** (Respecting solution). *Given a preclustered instance* $(\mathcal{K}, \chi_{\mathsf{pre}}, E_{\mathsf{adm}})$ *and a feasible solution* $(\mathcal{C}, \chi)$*, we say the solution* respects *the preclustered instance when*

- $\mathcal{K}$ *subdivides* $\mathcal{C}$*, i.e., for every precluster* $K \in \mathcal{K}$*, there exists a cluster* $C \in \mathcal{C}$ *such that* $K \subseteq C$*;*
- *for every non-singleton precluster* $K \in \mathcal{K}_{>1}$ *and the cluster* $C \in \mathcal{C}$ *containing* K*,* $\chi_{\mathsf{pre}}(K) = \chi(C)$*; and*
- *for every non-admissible edge* uv*, the two vertices* u *and* v *are not in a same cluster in the solution.*

In this section, we prove the following theorem.

▶ **Theorem 9.** *For any sufficiently small constant* $\varepsilon > 0$*, there exists an algorithm that, in time* $n^{\mathsf{poly}(\varepsilon^{-1})}$*, outputs a preclustered instance* $(\mathcal{K}, \chi_{\mathsf{pre}}, E_{\mathsf{adm}})$ *such that there exists a solution respecting the preclustered instance whose cost is at most* $(1+\varepsilon)\,\mathrm{opt}$*, and* $|E_{\mathsf{adm}}| \le O(\varepsilon^{-2})\,\mathrm{opt}$*, where* opt *denotes the cost of an optimal solution in the instance input to the algorithm.*

5.1 Constructing $\mathcal{K}$ and χ_{pre}

In this subsection, we present an algorithm for constructing a preclustering $\mathcal{K}$ and a color assignment $\chi_{\mathsf{pre}} : \mathcal{K}_{>1} \to L$. The algorithm is an adaptation of Cao et al.'s construction [9] to the chromatic versions. In particular, one subtlety in our algorithm is that we do not assume a self-loop for each vertex which is no loss of generality in the monochromatic version.

Algorithm description. We first describe the algorithm for CCC, and then state the changes needed for WCCC. We use parameters $\alpha, \beta \in (0,1)$ such that $\eta := \frac{\alpha+\beta}{1-\beta} \in (0, \frac{1}{13})$. Note that $\alpha = \beta = 0.02$ is one feasible choice.[2]

At the very start, we obtain a solution $(\mathcal{C}_{\mathsf{init}}, \chi_{\mathsf{init}})$ by running any constant-factor approximation algorithm for CCC on the given instance. We then iterate over every vertex $u \in V$ in an arbitrary order, and mark u if either

$$|E^{-\ell}(u, C)| \ge \alpha(|C| - 1) \text{ or } |E^{+}(u, V \setminus C)| \ge \alpha(|C| - 1) \text{ (or both)}, \tag{3}$$

where we denote by $C \in \mathcal{C}_{\mathsf{init}}$ the cluster containing u and by $\ell := \chi_{\mathsf{init}}(C)$ its color. After marking all such vertices, we now iterate over every cluster $C \in \mathcal{C}_{\mathsf{init}}$, and if the number of marked vertices in C is at least $\beta(|C| - 1)$, we mark all the vertices in C. We obtain $\mathcal{K}$ by making all marked vertices singleton clusters from $\mathcal{C}_{\mathsf{init}}$. The color assignment χ_{pre} is constructed by inheriting χ_{init}, i.e., for any non-singleton $K \in \mathcal{K}_{>1}$, we have $\chi_{\mathsf{pre}}(K) = \chi_{\mathsf{init}}(C)$, where $C \supseteq K$ is the cluster in $\mathcal{C}_{\mathsf{init}}$ containing K.

[2] The choice of α and β affects the constant hidden in the exponent of the running time $n^{\mathsf{poly}(\varepsilon^{-1})}$ for constructing a nearly optimal solution to the chromatic cluster LPs: this constant decreases as $\alpha\beta$ increases.

For WCCC, we only need to change the condition (3) of $u \in V$ being marked in the first iteration to

$w^{-\ell}(u, C) \geq \alpha(|C| - 1)$ or $w^{+}(u, V \setminus C) \geq \alpha(|C| - 1)$ (or both).

Recall that we present a constant-factor approximation algorithm for WCCC in Section 3.

Sketch of analysis. Let us now sketch the analysis of this algorithm. We focus on CCC while we mention the changes needed for WCCC at the end. The full analysis for CCC is given in Appendix B.1; while the proof for WCCC can be found in the full version.

We aim at proving the first two conditions of Definition 8 are satisfied by any optimal solution with the constructed $\mathcal{K}$ and χ_{pre}. Fix an arbitrary optimal solution $(\mathcal{C}^\star, \chi^\star)$. The next two lemmas are useful in the proof of the main lemmas of this subsection.

▶ **Lemma 10.** *For any non-singleton precluster $K \in \mathcal{K}_{>1}$ with $\ell := \chi_{\mathsf{pre}}(K)$ and $u \in K$, we have $|E^{-\ell}(u, K)| < \eta(|K| - 1)$ and $|E^{+}(u, V \setminus K)| < \eta(|K| - 1)$.*

▶ **Lemma 11.** *For any non-singleton precluster $K \in \mathcal{K}_{>1}$, if a cluster $C_1^\star \in \mathcal{C}^\star$ colored $\chi_{\mathsf{pre}}(K)$ intersects K (i.e., $\chi^\star(C_1^\star) = \chi_{\mathsf{pre}}(K)$ and $K \cap C_1^\star \neq \emptyset$), we have $|C_1^\star \setminus K| < 2\eta|K|$.*

Intuitively speaking, the former lemma follows from the execution of the algorithm in a fairly straightforward manner. The latter can be derived because, otherwise, splitting $C_1^\star$ into $C_1^\star \setminus K$ and $C_1^\star \cap K$ in $\mathcal{C}^\star$ incurs less cost due to the second condition of Lemma 10.

The following is the main technical lemma for the first condition of Definition 8. We provide a proof sketch of this lemma while the full proof can be found in Appendix B.1.

▶ **Lemma 12.** *$\mathcal{K}$ subdivides $\mathcal{C}^\star$, i.e., for any $K \in \mathcal{K}$, there is $C^\star \in \mathcal{C}^\star$ such that $K \subseteq C^\star$.*

Proof sketch. Suppose toward contradiction that $\mathcal{C}^\star$ is not subdivided by $\mathcal{K}$. Then, there exists a non-singleton precluster $K \in \mathcal{K}_{>1}$ such that $K \setminus C^\star$ is nonempty for every cluster $C^\star \in \mathcal{C}^\star$. We choose an arbitrary such precluster $K \in \mathcal{K}_{>1}$, and let $\ell := \chi_{\mathsf{pre}}(K)$ be the color of K in the preclustering. Let ξ be a constant between $\frac{1+5\eta}{2}$ and $1 - 4\eta$. Note that we can always choose such ξ because $\frac{1+5\eta}{2} < 1 - 4\eta$ for $\eta \in (0, \frac{1}{13})$. Observe that $\frac{1}{2} < \xi < 1$.

We break the proof into two cases depending on the existence of a cluster $C^\star \in \mathcal{C}^\star$ that "dominantly" intersects K, i.e., $|K \cap C^\star| \geq \xi|K|$. In each case, we respectively construct a new solution whose cost is strictly less than $(\mathcal{C}^\star, \chi^\star)$, leading to contradiction that $(\mathcal{C}^\star, \chi^\star)$ is optimal.

Case 1. No dominant intersections. Suppose that, for all $C^\star \in \mathcal{C}^\star$, $|K \cap C^\star| < \xi|K|$. In this case, we consider a new solution $(\mathcal{C}, \chi)$ by removing $K \cap C^\star$ from each $C^\star \in \mathcal{C}^\star$ and inserting K colored $\ell := \chi_{\mathsf{pre}}(K)$.

The "gain" of $(\mathcal{C}, \chi)$ over $(\mathcal{C}^\star, \chi^\star)$ is at least the total number of ℓ-colored edges within K but crossing $\mathcal{C}^\star$. For each $u \in K$, at least $(1 - \xi)|K|$ edges are crossing $\mathcal{C}^\star$ inside K due to the condition of this case, while at most $\eta|K|$ edges can be colored other than ℓ due to Lemma 10, implying that the gain is at least $\frac{1-\xi-\eta}{2}|K|^2$.

On the other hand, the "loss" of $(\mathcal{C}, \chi)$ over $(\mathcal{C}^\star, \chi^\star)$ is due to either the appropriately colored $+$ edges crossing K but within a cluster in $\mathcal{C}^\star$ or the $-$ edges within K but crossing $\mathcal{C}^\star$. By Lemma 10, we can bound this loss from above by $\frac{3\eta}{2}|K|^2 < \frac{1-\xi-\eta}{2}|K|^2$, where the inequality is due to the choice of ξ.

Case 2. A dominant intersection exists. Suppose now there exists a cluster $C_1^\star \in \mathcal{C}^\star$ such that $|K \cap C_1^\star| \geq \xi|K|$. Note that we always have at most one such cluster since $\xi > \frac{1}{2}$. Let $K_1 := K \cap C_1^\star$.

Assume for now that $\chi^\star(C_1^\star) = \chi_{\mathsf{pre}}(K)$. We construct a solution $(\mathcal{C}, \chi)$ by transferring one arbitrary vertex $u \in K \setminus C_1^\star$ to $C_1^\star$. We can then see that the gain of $(\mathcal{C}, \chi)$ over $(\mathcal{C}^\star, \chi^\star)$ is due to the ℓ-colored edges between u and $C_1^\star$, whereas the loss is due to the appropriately colored + edges incident with u within the cluster containing u in $\mathcal{C}^\star$ and edges other than ℓ-colored ones between u and $C_1^\star$. Using the dominance condition of this case and Lemma 10, we can show that the gain is at least $(\xi - \eta)|K|$.

To bound the loss from above, we prove that the number of the first type of edges is at most $(1 - \xi + \eta)|K|$ using the fact that $|K \setminus C_1^\star| \le (1 - \xi)|K|$ due to the case condition, and Lemma 10. On the other hand, the second type of edges can be partitioned into ones within K and ones incident with $C_1^\star \setminus K$; the former is bounded by $\eta|K|$ due to Lemma 10 while the latter is bounded by $2\eta|K|$ due to Lemma 11. Hence, the loss cannot exceed $(1 - \xi + 4\eta)|K| < (\xi - \eta)|K|$ by the choice of ξ.

Finally, for the case where $\chi^\star(C_1^\star) \neq \chi_{\mathsf{pre}}(K)$, we construct a solution $(\mathcal{C}, \chi)$ from $(\mathcal{C}^\star, \chi^\star)$ by splitting $C_1^\star$ into $K_1 := K \cap C_1^\star$ colored $\ell = \chi_{\mathsf{pre}}(K)$ and $C_1^\star \setminus K_1$ colored $\ell' := \chi^\star(C_1^\star)$. We observe that the gain of $(\mathcal{C}, \chi)$ against $(\mathcal{C}^\star, \chi^\star)$ is at least the number of the ℓ-colored edges within K_1 while the loss is at most the number of the ℓ'-colored edges within K_1 and between K_1 and $C_1^\star \setminus K_1$. Through a similar argument using Lemma 10, we can show that the gain is at least $\frac{1}{2}|K_1|(\xi|K| - 1 - \eta(|K| - 1))$ while the loss is at most $\frac{3}{2}\eta|K_1|(|K| - 1)$. The net gain is therefore at least $\frac{1}{2}|K_1| \cdot (\xi|K| - 1 - 4\eta(|K| - 1)) > 0$, where the inequality can be derived from the fact that K is non-singleton and the choice of ξ and η. ◀

The second condition of Definition 8 is also satisfied through a similar argument. The proof can be found in Appendix B.1.

▶ **Lemma 13.** *For every cluster $C^\star \in \mathcal{C}^\star$ and any non-singleton precluster $K \in \mathcal{K}$ contained in $C^\star$, we have $\chi_{\mathsf{pre}}(K) = \chi^\star(C^\star)$.*

For WCCC, we generalize the above analysis for CCC by substituting the number of edges satisfying a condition with the total sum of edge weights satisfying the same condition. In particular, instead of Lemma 10 for CCC, we have the following lemma for WCCC:

▶ **Lemma 14.** *For any non-singleton precluster $K \in \mathcal{K}_{>1}$ with $\ell := \chi_{\mathsf{pre}}(K)$ and $u \in K$, we have $w^{-\ell}(u, K) < \eta(|K| - 1)$ and $w^{+}(u, V \setminus K) < \eta(|K| - 1)$.*

Likewise, we have the counterparts of Lemmas 11, 12, and 13 for WCCC. The complete analysis can be found in the full version.

5.2 Constructing E_{adm}

We now present the construction of admissible edges E_{adm}. Interestingly, it turns out that a color-oblivious construction as in Cao et al. [9] suffices for the chromatic versions. Here we present its construction in the notation of WCCC.

For the unweighted version, it is noteworthy that the original construction [9] for the unweighted monochromatic version already incorporates weights. Hence, there exists a straightforward interpretation for CCC where we define $w^+(S, S') := |\{uv \in E^+ : u \in S, v \in S'\}|$ and $w^-(S, S') := |\{uv \in E^- : u \in S, v \in S'\}|$ for $S, S' \subseteq V$.

Given $\mathcal{K}$ from the previous section, the construction of E_{adm} is as follows. For $K \in \mathcal{K}$, let $d(K) := \frac{w^+(K, V \setminus K)}{|K|} + \frac{|K|}{2}$, $N_1(K) := \{K' \in \mathcal{K} \mid \varepsilon \cdot d(K') < d(K) < \frac{1}{\varepsilon} \cdot d(K')\}$, and $N_2(K) := \{K' \in \mathcal{K} \setminus \{K\} \mid K, K' \in N_1(K) \cap N_1(K')\}$. For distinct $K, K' \in \mathcal{K}$ and $\mathcal{K}' \subseteq \mathcal{K}$ such that $K, K' \in \mathcal{K}'$, we further define

$$W(K,K',\mathcal{K}') := \sum_{K''\in\mathcal{K}'\setminus\{K,K'\}} \left[|K''| \cdot \frac{w^+(K,K'')}{|K|\cdot|K''|} \cdot \frac{w^+(K',K'')}{|K'|\cdot|K''|}\right] + \frac{w^+(K,K')}{|K|} + \frac{w^+(K',K)}{|K'|}$$

Moreover, for two distinct preclusters $K, K' \in \mathcal{K}$, we denote by $K \sim K'$ if $K' \in N_2(K)$ and $W(K, K', N_1(K) \cap N_1(K')) > \varepsilon(d(K) + d(K'))$. We can then define $E_{\mathsf{adm}} := \{uv \in E \mid \exists K, K' \in \mathcal{K}$ such that $K \sim K', u \in K, v \in K'\}$.

This choice of E_{adm}, together with $\mathcal{K}$ and χ_{pre} from the previous section, satisfies the conditions of Theorem 9; the complete proof is given in the full version.

6 Solving the chromatic cluster LP

In this section, we present an algorithm for solving the chromatic cluster LP nearly optimally, using the preclustered instance $(\mathcal{K}, \chi_{\mathsf{pre}}, E_{\mathsf{adm}})$ satisfying Theorem 9. Our technique is obtained by introducing one more dimension of color to Cao et al.'s approach [9]. Recall that they first formulate the *bounded sub-cluster LP* making use of a constant-level Sherali-Adams hierarchy, present a procedure that samples one cluster using Raghavendra and Tan's rounding scheme [22], and lastly argue that the Monte Carlo method with sufficiently many samples from the procedure produces a nearly optimal solution with high probability.

In what follows, we describe the algorithm for solving the chromatic cluster LPs, adapted from Cao et al.'s approach to the chromatic versions. We remark that there are no differences in the algorithm and analysis between CCC and WCCC except the objective function of the bounded sub-cluster LP. The entire analysis is given in the full version.

Bounded sub-cluster LP. We first present the bounded sub-cluster LP for the chromatic versions. Given $\varepsilon > 0$ sufficiently small, we define $\varepsilon_1 := \varepsilon^3$, $\varepsilon_{\mathrm{rt}} := \varepsilon_1^2 = \varepsilon^6$, and $r := \Theta(\varepsilon_{\mathrm{rt}}^{-2}) = \Theta(\varepsilon^{-12})$. We assume that r is an integer where a large enough constant is hidden in its Θ-notation. For $u \in V$, let $K_u \in \mathcal{K}$ be the precluster containing u. For $u \in V$, let $N_{\mathsf{adm}}(u) := \{v \in V \mid uv \in E_{\mathsf{adm}}\}$; for $K \in \mathcal{K}$, let $N_{\mathsf{adm}}(K) := \{v \in V \mid \exists u \in K, uv \in E_{\mathsf{adm}}\}$. Then, the bounded sub-cluster LP is defined as follows:

$$\begin{aligned}
&\min \operatorname{obj}(x)\\
&\text{s.t. } y_S^\ell = \textstyle\sum_{s=1}^n y_S^{\ell,s}, && \forall \ell \in L\ \forall S \subseteq V : |S| \le r,\\
&\quad t_v^\ell = 1 - y_v^\ell, && \forall \ell \in L\ \forall v \in V,\\
&\quad x_{uv}^\ell = 1 - y_{uv}^\ell, && \forall \ell \in L\ \forall uv \in \tbinom{V}{2},\\
&\quad \textstyle\sum_{\ell\in L} y_v^\ell = 1, && \forall v \in V,\\
&\quad \textstyle\sum_{v\in V} y_{Sv}^{\ell,s} = s\, y_S^{\ell,s}, && \forall \ell \in L\ \forall s \in [n]\ \forall S \subseteq V : |S| \le r-1,\\
&\quad x_{uv}^{\chi_{\mathsf{pre}}(K)} = 0, && \forall uv \subseteq K \in \mathcal{K}_{>1},\\
&\quad x_{uv}^\ell = 1, && \forall uv \subseteq K \in \mathcal{K}_{>1}\ \forall \ell \ne \chi_{\mathsf{pre}}(K),\\
&\quad y_{uv}^{\ell,s} = y_u^{\ell,s}, && \forall \ell \in L\ \forall s \in [n]\ \forall u \in V\ \forall v \in K_u,\\
&\quad x_{uv}^\ell = 1, && \forall \text{non-admissible } uv\ \forall \ell \in L,\\
&\quad y_v^{\ell,s} = 0, && \\
&\qquad \forall \ell \in L\ \forall v \in V\ \forall s \in [|K_v| - 1] \cup [|K_v| + 1, |K_v| + \varepsilon_1 |N_{\mathsf{adm}}(v)|],
\end{aligned}$$

$$
\begin{aligned}
& y_S^{\chi_{\mathsf{pre}}(K),|K|} = y_v^{\chi_{\mathsf{pre}}(K),|K|}, && \forall v \in K \in \mathcal{K}_{>1}\ \forall S \subseteq K : |S| \le r, \\
& y_S^{\chi_{\mathsf{pre}}(K),|K|} = 0, && \forall K \in \mathcal{K}_{>1}\ \forall S \not\subseteq K : |S| \le r, S \cap K \neq \emptyset, \\
& \textstyle\sum_{T' \subseteq T} (-1)^{|T'|} y_{S \cup T'}^{\ell,s} \in [0, y_S^{\ell,s}], && \\
& \qquad \forall \ell \in L\ \forall s \in [n]\ \forall S, T \subseteq V : S \cap T = \emptyset, |S \cup T| \le r, && \\
& y_S^{\ell,s} \ge 0, && \forall \ell \in L\ \forall s \in [n]\ \forall S \subseteq V : |S| \le r,
\end{aligned}
$$

where $y_S^{\ell,s} = 1$ indicates whether S is a subset of a cluster of size s colored ℓ. The only distinction between CCC and WCCC is in the objective function, $\mathrm{obj}(x)$. Since the size of the bounded sub-cluster LP is indeed bounded by $n^{\mathsf{poly}(\varepsilon^{-1})}$, we can solve this LP in time $n^{\mathsf{poly}(\varepsilon^{-1})}$.

Sampling one colored cluster. We now present the procedure for sampling one cluster from the bounded sub-cluster LP. Let y be an optimal solution to the bounded sub-cluster LP. The procedure randomly samples a color $\ell \in L$ with probability $y_\emptyset^\ell / y_\emptyset$ where $y_\emptyset := \sum_{\ell' \in L} y_\emptyset^{\ell'}$, a cardinality $s \in [n]$ with probability $y_\emptyset^{\ell,s} / y_\emptyset^\ell$, and a pivot $u \in V$ with probability $y_u^{\ell,s} / (s \cdot y_\emptyset^{\ell,s})$. With these sampled ℓ, s, and u, define y' as $y'_S := y_{Su}^{\ell,s} / y_u^{\ell,s}$ for every $S \subseteq V$ of size $|S| \le r-1$. The procedure then runs Raghavendra and Tan's rounding scheme [22] upon y' to sample a cluster $C \subseteq V$, and return C colored $\chi(C) := \ell$.

Monte Carlo method. Finally, we describe the Monte Carlo method for constructing a nearly optimal solution. Let $\Delta := \Theta\left(\frac{n^4 \log n}{\varepsilon_1^2 |E_{\mathsf{adm}}|}\right)$ with a sufficiently large hidden constant. (Here, we assume $E_{\mathsf{adm}} \neq \emptyset$; otherwise, $(\mathcal{K}, \chi_{\mathsf{pre}})$ given by Theorem 9, with arbitrary colors assigned to singleton preclusters, is already a $(1+\varepsilon)$-approximate integral solution.) In particular, we assume that $\Delta y_\emptyset$ is an integer and $\Delta \ge \frac{1}{\varepsilon - \varepsilon_1} = \frac{1}{\varepsilon - \varepsilon^3}$. The method independently samples colored clusters from the above procedure for $\Delta y_\emptyset$ times. Let $C_1, \ldots, C_{\Delta y_\emptyset}$ be the sampled colored clusters. Let $\ell_1, \ldots, \ell_{\Delta y_\emptyset}$ denote their corresponding colors. For each vertex $v \in V$, let $R_v := \{t \in \{1, 2, \ldots, \Delta y_\emptyset\} \mid v \in C_t\}$, and let $\tilde{R}_v$ be the set of the $\lceil (1-\varepsilon)\Delta \rceil$ smallest indices in R_v. For every $t \in \{1, 2, \ldots, \Delta y_\emptyset\}$, we define $\tilde{C}_t := \{v \in V \mid t \in \tilde{R}_v\} \subseteq C_t$. The final output is $\tilde{z}_S^\ell := \frac{|\{t | \tilde{C}_t = S \wedge \ell_t = \ell\}|}{\lceil (1-\varepsilon)\Delta \rceil}$ for every $\ell \in L$ and $S \subseteq V$.

References

1 Fateme Abbasi, Hyung-Chan An, Jarosław Byrka, Changyeol Lee, and Yongho Shin. Approximating (weighted) chromatic correlation clustering via cluster LP. *arXiv preprint*, 2026. arXiv:2510.13446.

2 Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: ranking and clustering. *Journal of the ACM*, 55(5):1–27, 2008. doi:10.1145/1411509.1411513.

3 Ilya Amburg, Nate Veldt, and Austin Benson. Clustering in graphs and hypergraphs with categorical edge labels. In *Proceedings of The Web Conference (WWW)*, pages 706–717, 2020. doi:10.1145/3366423.3380152.

4 Yael Anava, Noa Avigdor-Elgrabli, and Iftah Gamzu. Improved theoretical and practical guarantees for chromatic correlation clustering. In *Proceedings of the 24th International Conference on World Wide Web (WWW)*, pages 55–65, 2015. doi:10.1145/2736277.2741629.

5 Eric Angel, Evripidis Bampis, A. Kononov, Dimitris Paparas, Emmanouil Pountourakis, and Vassilis Zissimopoulos. Clustering on k-edge-colored graphs. *Discrete Applied Mathematics*, 211:15–22, 2016. doi:10.1016/J.DAM.2016.04.017.

6 Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56:89–113, 2004. doi:10.1023/B:MACH.0000033116.57574.95.

7 Amir Ben-Dor and Zohar Yakhini. Clustering gene expression patterns. In *Proceedings of the Third Annual International Conference on Computational Molecular Biology (RECOMB)*, pages 33–42, 1999. doi:10.1145/299432.299448.

8 Francesco Bonchi, Aristides Gionis, Francesco Gullo, Charalampos E. Tsourakakis, and Antti Ukkonen. Chromatic correlation clustering. *ACM Transactions on Knowledge Discovery from Data*, 9(4):1–24, 2015. doi:10.1145/2728170.

9 Nairen Cao, Vincent Cohen-Addad, Euiwoong Lee, Shi Li, Alantha Newman, and Lukas Vogl. Understanding the cluster linear program for correlation clustering. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1605–1616, 2024. doi:10.1145/3618260.3649749.

10 Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. *Journal of Computer and System Sciences*, 71(3):360–383, 2005. doi:10.1016/J.JCSS.2004.10.012.

11 Shuchi Chawla, Konstantin Makarychev, Tselil Schramm, and Grigory Yaroslavtsev. Near optimal LP rounding algorithm for correlation clustering on complete and complete k-partite graphs. In *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing (STOC)*, pages 219–228, 2015. doi:10.1145/2746539.2746604.

12 Zhi-Zhong Chen, Tao Jiang, and Guohui Lin. Computing phylogenetic roots with bounded degrees and errors. *SIAM Journal on Computing*, 32(4):864–879, 2003. doi:10.1137/S0097539701389154.

13 Vincent Cohen-Addad, Euiwoong Lee, Shi Li, and Alantha Newman. Handling correlated rounding error via preclustering: A 1.73-approximation for correlation clustering. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1082–1104. IEEE, 2023. doi:10.1109/FOCS57990.2023.00065.

14 Vincent Cohen-Addad, Euiwoong Lee, and Alantha Newman. Correlation clustering with Sherali-Adams. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 651–661. IEEE, 2022. doi:10.1109/FOCS54457.2022.00068.

15 Patrick Doreian and Andrej Mrvar. A partitioning approach to structural balance. *Social Networks*, 18(2):149–168, 1996.

16 Chenglin Fan, Dahoon Lee, and Euiwoong Lee. 1.64-approximation for chromatic correlation clustering via chromatic cluster LP. In *Integer Programming and Combinatorial Optimization (IPCO)*, pages 81–96. Springer Nature Switzerland, 2026. doi:10.1007/978-3-032-28691-8_6.

17 Aristides Gionis, Heikki Mannila, and Panayiotis Tsaparas. Clustering aggregation. *ACM Transactions on Knowledge Discovery from Data*, 1(1):4–es, 2007.

18 Nicolas Klodt, Lars Seifert, Arthur Zahn, Katrin Casel, Davis Issac, and Tobias Friedrich. A color-blind 3-approximation for chromatic correlation clustering and improved heuristics. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (KDD)*, pages 882–891, 2021. doi:10.1145/3447548.3467446.

19 Dahoon Lee, Chenglin Fan, and Euiwoong Lee. 1.64-approximation for chromatic correlation clustering via chromatic cluster LP. *arXiv preprint*, 2025. doi:10.48550/arXiv.2507.15417.

20 Dahoon Lee, Chenglin Fan, and Euiwoong Lee. Improved approximation algorithms for chromatic and pseudometric-weighted correlation clustering. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

21 Mojtaba Ostovari and Alireza Zarei. Improved combinatorial approximations for weighted correlation clustering. *Discrete Mathematics & Theoretical Computer Science*, 27(Discrete Algorithms), 2025. doi:10.46298/DMTCS.12424.

22 Prasad Raghavendra and Ning Tan. Approximating CSPs with global cardinality constraints using SDP hierarchies. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 373–387. SIAM, 2012. doi:10.1137/1.9781611973099.33.

23 S. Regnie. On some mathematical aspects of automatic classification problems. *Math and Human Sciences*, 82:13–29, 1965.

24 Ron Shamir, Roded Sharan, and Dekel Tsur. Cluster graph modification problems. *Discrete Applied Mathematics*, 144(1-2):173–182, 2004. doi:10.1016/J.DAM.2004.01.007.

25 Anke van Zuylen and David P. Williamson. Deterministic pivoting algorithms for constrained ranking and clustering problems. *Mathematics of Operations Research*, 34(3):594–620, 2009. doi:10.1287/MOOR.1090.0385.

26 Dewan F. Wahid and Elkafi Hassini. A literature review on correlation clustering: Cross-disciplinary taxonomy with bibliometric analysis. *Operations Research Forum*, 3(3), 2022. doi:10.1007/S43069-022-00156-6.

27 Qing Xiu, Kai Han, Jing Tang, Shuang Cui, and He Huang. Chromatic correlation clustering, revisited. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:26147–26159, 2022.

28 Charles T. Zahn, Jr. Approximating symmetric relations by equivalence relations. *Journal of the Society for Industrial and Applied Mathematics*, 12(4):840–847, 1964.

A Deferred analysis from Section 4

We can rephrase the objective function for CCC as follows:

$$
\begin{aligned}
&\textstyle\sum_{S\subseteq V,\ell\in L}\left(\frac{|\delta^+(S)|}{2}+|E^{-\ell}[S]|\right)z_S^\ell\\
&=\textstyle\sum_{uv\in E^+}\left[\frac{1}{2}\sum_{\ell\in L}\left(\sum_{S\ni u,S\not\ni v}z_S^\ell+\sum_{S\not\ni u,S\ni v}z_S^\ell\right)+\sum_{S\supseteq uv,\ell\neq c_{uv}}z_S^\ell\right]\\
&\quad+\textstyle\sum_{uv\in E^-}\sum_{\ell\in L}\sum_{S\supseteq uv}z_S^\ell\\
&=\textstyle\sum_{uv\in E^+}\left[\sum_{\ell\in L}x_{uv}^\ell-\sum_{\ell\in L}\frac{t_u^\ell+t_v^\ell}{2}+\sum_{\ell\neq c_{uv}}\sum_{S\supseteq uv}z_S^\ell\right]\\
&\quad+\textstyle\sum_{uv\in E^-}\sum_{\ell\in L}\sum_{S\supseteq uv}z_S^\ell\\
&=\textstyle\sum_{uv\in E^+}\left[\sum_{\ell\in L}x_{uv}^\ell-(|L|-1)+\sum_{\ell\neq c_{uv}}(1-x_{uv}^\ell)\right]\\
&\quad+\textstyle\sum_{uv\in E^-}\sum_{\ell\in L}\sum_{S\supseteq uv}z_S^\ell\\
&=\textstyle\sum_{uv\in E^+}x_{uv}^{c_{uv}}+\sum_{uv\in E^-}\sum_{\ell\in L}(1-x_{uv}^\ell)=:\mathrm{obj}(x),
\end{aligned}
$$

where the second equality comes from (1) and the third from (2).

Using a similar derivation, we can also rephrase the objective function for WCCC as follows:

$$
\begin{aligned}
&\sum_{S\subseteq V,\ell\in L}\left(\frac{w^+(S,V\setminus S)}{2}+w^{-\ell}(S,S)\right)z_S^\ell\\
&=\sum_{uv\in E}\left[\frac{w^+(u,v)}{2}\sum_{\ell\in L}\left(\sum_{S\ni u,S\not\ni v}z_S^\ell+\sum_{S\not\ni u,S\ni v}z_S^\ell\right)+\sum_{S\supseteq uv}\sum_{\ell\in L}w^{-\ell}(u,v)\cdot z_S^\ell\right]\\
&=\sum_{uv\in E}\left[w^+(u,v)\cdot\left(\sum_{\ell\in L}x_{uv}^\ell-\sum_{\ell\in L}\frac{t_u^\ell+t_v^\ell}{2}\right)+\sum_{S\supseteq uv}\sum_{\ell\in L}w^{-\ell}(u,v)\cdot z_S^\ell\right]\\
&=\sum_{uv\in E}\left[w^+(u,v)\cdot\left(\sum_{\ell\in L}x_{uv}^\ell-(|L|-1)\right)+\sum_{\ell\in L}w^{-\ell}(u,v)\cdot(1-x_{uv}^\ell)\right]\\
&=\sum_{uv\in E}\left[\sum_{\ell\in L}(w^+(u,v)-w^{-\ell}(u,v))\cdot x_{uv}^\ell-w^+(u,v)(|L|-1)+\sum_{\ell\in L}w^{-\ell}(u,v)\right]\\
&=\sum_{uv\in E}\left[\sum_{\ell\in L}(w^\ell(u,v)-w^-(u,v))\cdot x_{uv}^\ell+w^-(u,v)|L|\right]
\end{aligned}
$$

$$= \sum_{uv \in E} \sum_{\ell \in L} \left(w^{\ell}(u,v) x^{\ell}_{uv} + w^{-}(u,v)(1 - x^{\ell}_{uv}) \right)$$
$$=: \mathrm{obj}(x),$$

where the second equality comes from (1), the third from (2), and the fifth from the definition of w^{+} and $w^{-\ell}$.

Proof of Lemma 4. Consider the set S such that $S \cap \{u, v\} \neq \emptyset$ for the first time. We have

$$\begin{aligned} \textstyle\sum_{\ell' \in L, S : S \cap uv \neq \emptyset} z^{\ell'}_S &= \textstyle\sum_{\ell' \in L, S \ni u} z^{\ell'}_S + \sum_{\ell' \in L, S \ni v} z^{\ell'}_S - \sum_{\ell' \in L, S \supseteq uv} z^{\ell'}_S \\ &= 1 + x_{uv}. \end{aligned}$$

For the first statement, recall that $\sum_{S \supseteq uv} z^{\ell}_S = 1 - x^{\ell}_{uv}$. The second statement immediately follows from the fact that $\sum_{\ell' \in L, S \supseteq uv} z^{\ell'}_S = 1 - x_{uv}$. ◀

B Deferred analysis from Section 5

B.1 Constructing $\mathcal{K}$ and χ_{pre} for CCC

Recall that we use parameters $\alpha, \beta \in (0,1)$ such that $\eta := \frac{\alpha+\beta}{1-\beta} \in (0, \frac{1}{13})$.

▶ **Lemma 15** (Restatement of Lemma 10). *For any non-singleton precluster $K \in \mathcal{K}_{>1}$ with $\ell := \chi_{\mathsf{pre}}(K)$ and $u \in K$, we have*

$$|E^{-\ell}(u, K)| < \eta(|K| - 1) \text{ and } |E^{+}(u, V \setminus K)| < \eta(|K| - 1).$$

Proof. Let $C \in \mathcal{C}_{\mathsf{init}}$ be the cluster containing $K \subseteq C$. Since K is non-singleton, the vertices in K remain unmarked during the execution of the algorithm, implying that

$$|C \setminus K| < \beta(|C| - 1). \tag{4}$$

Since $|C \setminus K| = (|C| - 1) - (|K| - 1)$, rearranging the terms yields

$$|C| - 1 < \frac{1}{1-\beta}(|K| - 1). \tag{5}$$

Let $\ell := \chi_{\mathsf{pre}}(K) = \chi_{\mathsf{init}}(C)$. We can also observe that

$$|E^{-\ell}(u, C)| < \alpha(|C| - 1) \text{ and } |E^{+}(u, V \setminus C)| < \alpha(|C| - 1) \tag{6}$$

since $u \in K$ is unmarked. We therefore have

$$|E^{-\ell}(u, K)| \leq |E^{-\ell}(u, C)| < \alpha(|C| - 1) < \frac{\alpha}{1-\beta}(|K| - 1) < \eta(|K| - 1),$$

where the first inequality follows from the fact that $E^{-\ell}(u, K) \subseteq E^{-\ell}(u, C)$ and the second-to-last inequality is due to Inequality (5). Furthermore, we have

$$\begin{aligned} |E^{+}(u, V \setminus K)| &= |E^{+}(u, C \setminus K)| + |E^{+}(u, V \setminus C)| \leq |C \setminus K| + |E^{+}(u, V \setminus C)| \\ &< \beta(|C| - 1) + \alpha(|C| - 1) < \frac{\alpha + \beta}{1 - \beta}(|K| - 1) = \eta\,(|K| - 1), \end{aligned}$$

where the second inequality is derived from Inequalities (4) and (6) and the last from Inequality (5). ◀

In what follows, we may instead use the following slightly weaker inequalities in most cases:

$$|E^{-\ell}(u, K)| < \eta|K| \text{ and } |E^{+}(u, V \setminus K)| < \eta|K|.$$

Now, fix any optimal solution $(\mathcal{C}^\star, \chi^\star)$ to the given instance I.

▶ **Lemma 16** (Restatement of Lemma 11 for CCC). *For any non-singleton precluster $K \in \mathcal{K}_{>1}$, if a cluster $C_1^\star \in \mathcal{C}^\star$ colored $\chi_{\mathsf{pre}}(K)$ intersects K (i.e., $\chi^\star(C_1^\star) = \chi_{\mathsf{pre}}(K)$ and $K \cap C_1^\star \neq \emptyset$), we have $|C_1^\star \setminus K| < 2\eta|K|$.*

Proof. Let $\ell := \chi_{\mathsf{pre}}(K) = \chi^\star(C_1^\star)$ and $K_1 := K \cap C_1^\star$. We define a solution $(\mathcal{C}, \chi)$ constructed from $(\mathcal{C}^\star, \chi^\star)$ by splitting $C_1^\star$ into K_1 and $C_1^\star \setminus K_1$. That is,

$$\mathcal{C} := \mathcal{C}^\star \setminus \{C_1^\star\} \cup \{K_1, C_1^\star \setminus K_1\} \text{ and } \chi(C) := \begin{cases} \ell, & \text{if } C \in \{K_1, C_1^\star \setminus K_1\}, \\ \chi^\star(C), & \text{if } C \in \mathcal{C} \cap \mathcal{C}^\star. \end{cases}$$

Suppose toward contradiction that $|C_1^\star \setminus K| \geq 2\eta|K|$. We show that the number of disagreements in $(\mathcal{C}, \chi)$ is less than that of $(\mathcal{C}^\star, \chi^\star)$. To this end, we classify all edges as follows, and determine whether each edge is a disagreement or not in $(\mathcal{C}, \chi)$ and in $(\mathcal{C}^\star, \chi^\star)$:

1. Each $-$ edge across K_1 and $C_1^\star \setminus K_1$ is an agreement in $(\mathcal{C}, \chi)$, but a disagreement in $(\mathcal{C}^\star, \chi^\star)$.
2. Each $-$ edge across K_1 and $V \setminus C_1^\star$ is an agreement in both solutions.
3. Each $+$ edge colored ℓ across K_1 and $C_1^\star \setminus K_1$ is a disagreement in $(\mathcal{C}, \chi)$, but an agreement in $(\mathcal{C}^\star, \chi^\star)$.
4. Each $+$ edge colored differently from ℓ across K_1 and $C_1^\star \setminus K_1$ is a disagreement in both solutions.
5. Each $+$ edge across K_1 and $V \setminus C_1^\star$ is a disagreement in both solutions.
6. Each edge within K_1 has the same status in both solutions.
7. The remaining edges outside $C_1^\star$ also have the same statuses in both solutions, respectively.

We first lower bound the "gain" in cost of the constructed solution $(\mathcal{C}, \chi)$ – the number of edges that are agreed in $(\mathcal{C}, \chi)$, but not in $(\mathcal{C}^\star, \chi^\star)$. Observe that the type 1 edges contribute to this gain. For any $u \in K_1$, note that

$$\begin{aligned} |E^{-}(u, C_1^\star \setminus K_1)| &= |E(u, C_1^\star \setminus K_1)| - |E^{+}(u, C_1^\star \setminus K_1)| \\ &= |E(u, C_1^\star \setminus K_1)| - |E^{+}(u, C_1^\star \setminus K)| \\ &\geq |E(u, C_1^\star \setminus K_1)| - |E^{+}(u, V \setminus K)| \\ &= |C_1^\star \setminus K_1| - |E^{+}(u, V \setminus K)| \\ &> |C_1^\star \setminus K_1| - \eta|K|, \end{aligned}$$

where the last equality comes from $u \notin C_1^\star \setminus K_1$, and the last inequality from Lemma 15 (recall that K is non-singleton). We can thus derive that the gain of $(\mathcal{C}, \chi)$, the total number of type 1 edges, is strictly greater than $|K_1| \cdot (|C_1^\star \setminus K_1| - \eta|K|) = |K_1| \cdot (|C_1^\star \setminus K| - \eta|K|)$.

We now upper bound the "loss" in cost of $(\mathcal{C}, \chi)$ – the number of edges that are agreed in $(\mathcal{C}^\star, \chi^\star)$, but not in $(\mathcal{C}, \chi)$. Note that the edges of type 3 contribute to this loss. For any $u \in K_1$, we have

$$|E^{\ell}(u, C_1^\star \setminus K_1)| = |E^{\ell}(u, C_1^\star \setminus K)| \leq |E^{+}(u, V \setminus K)| < \eta|K|$$

where the last inequality is again due to Lemma 15, yielding that the loss of $(\mathcal{C}, \chi)$ is strictly less than $|K_1| \cdot \eta|K|$.

We can thus conclude that the "net gain" in cost of the constructed solution $(\mathcal{C}, \chi)$ against $(\mathcal{C}^\star, \chi^\star)$ is strictly greater than

$$|K_1| \cdot (|C_1^\star \setminus K| - \eta|K|) - |K_1| \cdot \eta|K| = |K_1| \cdot (|C_1^\star \setminus K| - 2\eta|K|) \geq 0,$$

where the inequality is due to the assumption that $|C_1^\star \setminus K| \geq 2\eta|K|$, contradicting the optimality of $(\mathcal{C}^\star, \chi^\star)$. ◀

▶ **Lemma 17** (Restatement of Lemma 12). *$\mathcal{K}$ subdivides $\mathcal{C}^\star$, i.e., for any $K \in \mathcal{K}$, there is $C^\star \in \mathcal{C}^\star$ such that $K \subseteq C^\star$.*

Proof. Suppose toward contradiction that $\mathcal{C}^\star$ is not subdivided by $\mathcal{K}$. Then, there exists a non-singleton precluster $K \in \mathcal{K}_{>1}$ such that $K \setminus C^\star$ is nonempty for every cluster $C^\star \in \mathcal{C}^\star$. We choose an arbitrary such precluster $K \in \mathcal{K}_{>1}$, and let $\ell := \chi_{\mathsf{pre}}(K)$ be the color of K in the preclustering. Let ξ be a constant between $\frac{1+5\eta}{2}$ and $1 - 4\eta$. Note that we can always choose such ξ because $\frac{1+5\eta}{2} < 1 - 4\eta$ for $\eta \in (0, \frac{1}{13})$. Observe that $\frac{1}{2} < \xi < 1$.

We break the analysis into two cases depending on the existence of a cluster $C^\star \in \mathcal{C}^\star$ that "dominantly" intersects K, i.e., $|K \cap C^\star| \geq \xi|K|$. The overall structure of the proof for each case is similar to the proof of Lemma 16: we respectively construct a solution whose cost is strictly less than $(\mathcal{C}^\star, \chi^\star)$, leading to contradiction that $(\mathcal{C}^\star, \chi^\star)$ is optimal.

Case 1. No dominant intersections. Suppose that, for all $C^\star \in \mathcal{C}^\star$, $|K \cap C^\star| < \xi|K|$. Consider a new solution $(\mathcal{C}', \chi')$ constructed by inserting K colored $\ell = \chi_{\mathsf{pre}}(K)$. More precisely, we have

$$\mathcal{C}' := \{K\} \cup \{C^\star \setminus K \mid C^\star \in \mathcal{C}^\star \text{ with } |C^\star \setminus K| > 0\} \text{ and}$$
$$\chi'(C) := \begin{cases} \ell, & \text{if } C = K; \\ \chi^\star(C^\star), & \text{if } C = C^\star \setminus K. \end{cases}$$

We classify each edge and determine whether it is a disagreement or not in both solutions.

1. Each $-$ edge across K and $V \setminus K$ is agreed in $(\mathcal{C}', \chi')$, and possibly agreed in $(\mathcal{C}^\star, \chi^\star)$.
2. Each $+$ edge across K and $V \setminus K$ is disagreed in $(\mathcal{C}', \chi')$, but possibly agreed in $(\mathcal{C}^\star, \chi^\star)$.
3. Each $-$ edge within K is disagreed in $(\mathcal{C}', \chi')$, but possibly agreed in $(\mathcal{C}^\star, \chi^\star)$.
4. Each $+$ edge e of color $c_e = \ell$ within $K \cap C^\star$ for some $C^\star \in \mathcal{C}^\star$ is agreed in $(\mathcal{C}', \chi')$, and possibly agreed in $(\mathcal{C}^\star, \chi^\star)$.
5. Each $+$ edge e of color $c_e \neq \ell$ within $K \cap C^\star$ for some $C^\star \in \mathcal{C}^\star$ is disagreed in $(\mathcal{C}', \chi')$, but possibly agreed in $(\mathcal{C}^\star, \chi^\star)$.
6. Each $+$ edge e of color $c_e = \ell$ that is within K but across clusters in $\mathcal{C}^\star$ is agreed in $(\mathcal{C}', \chi')$, but disagreed in $(\mathcal{C}^\star, \chi^\star)$.
7. Each $+$ edge e of color $c_e \neq \ell$ that is within K but across clusters in $\mathcal{C}^\star$ is disagreed in both solutions.
8. The remaining edges outside K have the same statuses in both solutions, respectively.

Observe that the gain in cost of $(\mathcal{C}', \chi')$ against $(\mathcal{C}^\star, \chi^\star)$ is lower-bounded by the number of type 6 edges while the loss in cost of $(\mathcal{C}', \chi')$ is upper-bounded by the number of edges of types 2, 3, and 5.

To give a lower bound on the number of type 6 edges, we consider any $u \in K$ and let $C_u^\star \in \mathcal{C}^\star$ be the cluster containing u in $\mathcal{C}^\star$. We then have

$$\begin{aligned} |E^\ell(u, K \setminus C_u^\star)| &= |E(u, K \setminus C_u^\star)| - |E^{-\ell}(u, K \setminus C_u^\star)| \\ &\geq |E(u, K \setminus C_u^\star)| - |E^{-\ell}(u, K)| \\ &= |K \setminus C_u^\star| - |E^{-\ell}(u, K)| > |K \setminus C_u^\star| - \eta|K| \\ &\geq (1 - \xi)|K| - \eta|K| = (1 - \xi - \eta)|K|, \end{aligned}$$

where the second equality comes from $u \notin K \setminus C_u^\star$, the second inequality from Lemma 15, and the last inequality from the condition of this case. Hence, the number of type 6 edges is bounded from below by

$$\frac{1}{2}\sum_{u\in K}|E^{\ell}(u, K\setminus C_u^\star)| > \frac{1-\xi-\eta}{2}|K|^2.$$

This is also a lower bound for the gain in cost of $(\mathcal{C}', \chi')$ against $(\mathcal{C}^\star, \chi^\star)$.

We now consider the edges of types 2, 3 and 5. Due to the second inequality of Lemma 15, we can upper bound the number of type 2 edges by $|K|\cdot\eta|K| = \eta|K|^2$. On the other hand, the total number of type 3 and type 5 edges is bounded from above by

$$\frac{1}{2}\sum_{u\in K}|E^{-\ell}(u, K)| < \frac{\eta}{2}|K|^2$$

due to the first inequality of Lemma 15. Hence, the loss in cost of $(\mathcal{C}', \chi')$ against $(\mathcal{C}^\star, \chi^\star)$ is strictly less than $\frac{3\eta}{2}|K|^2$. Since $\xi \le 1-4\eta$, we have

$$\frac{1-\xi-\eta}{2}|K|^2 - \frac{3\eta}{2}|K|^2 \ge 0,$$

implying that $(\mathcal{C}', \chi')$ has fewer disagreements than $(\mathcal{C}^\star, \chi^\star)$. This contradicts the optimality of $(\mathcal{C}^\star, \chi^\star)$.

Case 2. A dominant intersection exists. Suppose now there exists a cluster $C_1^\star \in \mathcal{C}^\star$ such that $|K\cap C_1^\star| \ge \xi|K|$. Note that we always have at most one such cluster since $\xi > \frac{1}{2}$. Let $K_1 := K \cap C_1^\star$.

We first claim that $\chi^\star(C_1^\star) \ne \chi_{\mathsf{pre}}(K)$. Otherwise, if $\chi^\star(C_1^\star) = \chi_{\mathsf{pre}}(K)$, we construct a solution $(\mathcal{C}'', \chi'')$ by transferring one arbitrary vertex $w \in K \setminus C_1^\star$ to $C_1^\star$. In other words,

$$\mathcal{C}'' := \{C_1^\star \cup \{w\}\} \cup \{C^\star \setminus \{w\} \mid C^\star \in \mathcal{C}^\star \setminus \{C_1^\star\}\} \text{ and}$$
$$\chi''(C) = \begin{cases} \chi^\star(C_1^\star), & \text{if } C = C_1^\star \cup \{w\}, \\ \chi^\star(C^\star), & \text{if } C = C^\star \setminus \{w\}. \end{cases}$$

Since $K \setminus C_1^\star$ is nonempty, we can always choose $w \in K \setminus C_1^\star$ in the above construction. Recall also that $\ell = \chi_{\mathsf{pre}}(K) = \chi^\star(C_1^\star)$. Consider the following classification of edges:

1. Each edge in $E^{\ell}(w, C_1^\star)$ is agreed in $(\mathcal{C}'', \chi'')$, but disagreed in $(\mathcal{C}^\star, \chi^\star)$.
2. Each edge in $E^{-\ell}(w, C_1^\star)$ is disagreed in $(\mathcal{C}'', \chi'')$, but possibly agreed in $(\mathcal{C}^\star, \chi^\star)$.
3. Each edge in $E^{+}(w, V\setminus C_1^\star)$ is disagreed in $(\mathcal{C}'', \chi'')$, but possibly agreed in $(\mathcal{C}^\star, \chi^\star)$.
4. Each edge in $E^{-}(w, V\setminus C_1^\star)$ is agreed in $(\mathcal{C}'', \chi'')$, and possibly agreed in $(\mathcal{C}^\star, \chi^\star)$.
5. All the edges that are not incident with w have the same statuses in both solutions, respectively.

Hence, the gain in cost of $(\mathcal{C}'', \chi'')$ against $(\mathcal{C}^\star, \chi^\star)$ is bounded from below by $|E^{\ell}(w, C_1^\star)|$, whilst the loss in cost of $(\mathcal{C}'', \chi'')$ is bounded from above by $|E^{-\ell}(w, C_1^\star)| + |E^{+}(w, V\setminus C_1^\star)|$. For the type 1 edges, we have

$$\begin{aligned} |E^{\ell}(w, C_1^\star)| &\ge |E^{\ell}(w, K_1)| = |E(w, K_1)| - |E^{-\ell}(w, K_1)| \\ &= |K_1| - |E^{-\ell}(w, K_1)| \ge |K_1| - |E^{-\ell}(w, K)| \\ &> |K_1| - \eta|K| \ge \xi|K| - \eta|K| = (\xi-\eta)|K|, \end{aligned}$$

where the second equality comes from $w \notin K_1$, and the second-to-last inequality from Lemma 15. For type 2 edges,

$$\begin{aligned}|E^{-\ell}(w, C_1^\star)| &= |E^{-\ell}(w, K_1)| + |E^{-\ell}(w, C_1^\star \setminus K_1)| \\ &\le |E^{-\ell}(w, K)| + |C_1^\star \setminus K_1| \\ &= |E^{-\ell}(w, K)| + |C_1^\star \setminus K| \\ &< \eta|K| + 2\eta|K| = 3\eta|K|,\end{aligned}$$

where the last inequality comes from Lemma 15 and Lemma 16. Lastly, for type 3 edges,

$$\begin{aligned}|E^{+}(w, V \setminus C_1^\star)| &= |E^{+}(w, K \setminus C_1^\star)| + |E^{+}(w, (V \setminus K) \setminus C_1^\star)| \\ &< |K \setminus C_1^\star| + |E^{+}(w, V \setminus K)| \\ &\le (1-\xi)|K| + |E^{+}(w, V \setminus K)| \\ &< (1-\xi)|K| + \eta|K| = (1-\xi+\eta)|K|,\end{aligned}$$

where the second inequality comes from that $|K \cap C_1^\star| \ge \xi|K|$, and the last inequality is due to Lemma 15. Hence, the net gain in cost of $(\mathcal{C}'', \chi'')$ against $(\mathcal{C}^\star, \chi^\star)$ is strictly greater than

$$(\xi - \eta)|K| - (1 - \xi + 4\eta)|K| \ge 0,$$

where the inequality follows from the choice of $\xi > \frac{1+5\eta}{2}$, contradicting the optimality of $(\mathcal{C}^\star, \chi^\star)$. This completes the proof of our claim that $\chi^\star(C_1^\star) \neq \chi_{\mathsf{pre}}(K)$.

We now construct another solution $(\mathcal{C}, \chi)$ from $(\mathcal{C}^\star, \chi^\star)$ by splitting $C_1^\star$ into K_1 colored $\ell = \chi_{\mathsf{pre}}(K)$ and $C_1^\star \setminus K_1$ colored $\chi^\star(C_1^\star)$, i.e.,

$$\mathcal{C} := \mathcal{C}^\star \setminus \{C_1^\star\} \cup \{K_1, C_1^\star \setminus K_1\} \quad \text{and} \quad \chi(C) := \begin{cases} \ell, & \text{if } C = K_1, \\ \chi^\star(C_1^\star), & \text{if } C = C_1^\star \setminus K_1, \\ \chi^\star(C), & \text{if } C \in \mathcal{C} \cap \mathcal{C}^\star. \end{cases}.$$

Consider the following breakdown of the edges:

1. Each $-$ edge across K_1 and $V \setminus K_1$ is agreed in $(\mathcal{C}, \chi)$, and possibly agreed in $(\mathcal{C}^\star, \chi^\star)$.
2. Each $+$ edge across K_1 and $C_1^\star \setminus K_1$ is disagreed in $(\mathcal{C}, \chi)$, but possibly agreed in $(\mathcal{C}^\star, \chi^\star)$.
3. Each $+$ edge across K_1 and $V \setminus C_1^\star$ is disagreed in both solutions.
4. Each $-$ edge within K_1 is disagreed in both solutions.
5. Each $+$ edge e of color $c_e = \ell$ within K_1 is agreed in $(\mathcal{C}, \chi)$, but disagreed in $(\mathcal{C}^\star, \chi^\star)$.
6. Each $+$ edge e of color $c_e \neq \ell$ within K_1 is disagreed in $(\mathcal{C}, \chi)$, but possibly agreed in $(\mathcal{C}^\star, \chi^\star)$.
7. All the remaining edges outside K_1 have the same statuses in both solutions, respectively.

We can thus see that the gain in cost of $(\mathcal{C}, \chi)$ against $(\mathcal{C}^\star, \chi^\star)$ is at least the number of type 5 edges, while the loss is at most the total number of type 2 and type 6 edges.

To lower-bound the number of type 5 edges, we first fix an $u \in K_1$ and consider the type 5 edges incident to u. We then have

$$\begin{aligned}|E^{\ell}(u, K_1)| &= |E(u, K_1)| - |E^{-\ell}(u, K_1)| = |K_1| - 1 - |E^{-\ell}(u, K_1)| \\ &\ge |K_1| - 1 - |E^{-\ell}(u, K)| > |K_1| - 1 - \eta(|K| - 1) \\ &\ge \xi|K| - 1 - \eta(|K| - 1),\end{aligned}$$

where the second inequality comes from Lemma 15, and the last inequality comes from the condition of K_1. Hence, the number of type 5 edges is at least

$$\frac{1}{2}\sum_{u\in K_1}|E^{\ell}(u,K_1)| > \frac{1}{2}|K_1|\cdot(\xi|K|-1-\eta(|K|-1)).$$

We now upper-bound the number of edges of types 2 and 6. For any $u \in K_1$, we have

$$|E^{+}(u,C_1^{\star}\setminus K_1)| = |E^{+}(u,C_1^{\star}\setminus K)| \le |E^{+}(u,V\setminus K)| < \eta(|K|-1),$$

where the last inequality is due to Lemma 15, implying that the number of type 2 edges is strictly less than $|K_1|\cdot\eta(|K|-1)$. On the other hand, we also have that, for every $u \in K_1$,

$$|E^{+}\cap E^{-\ell}(u,K_1)| \le |E^{-\ell}(u,K)| < \eta(|K|-1)$$

where the last inequality comes from Lemma 15, yielding that the number of type 6 edges is strictly upper-bounded by $\frac{1}{2}|K_1|\cdot\eta(|K|-1)$.

We can thus derive that the net gain in cost of $(\mathcal{C},\chi)$ against $(\mathcal{C}^{\star},\chi^{\star})$ is greater than

$$\frac{1}{2}|K_1|\cdot(\xi|K|-1-\eta(|K|-1)) - \frac{1}{2}|K_1|\cdot 3\eta(|K|-1)$$
$$= \frac{1}{2}|K_1|\cdot(\xi|K|-1-4\eta(|K|-1)).$$

Observe that $\xi > \frac{1+5\eta}{2} > \frac{1}{2} > \frac{4}{13} > 4\eta$ by the choice of ξ and η. We can thus see

$$\xi|K|-1-4\eta(|K|-1) = (\xi-4\eta)|K|-1+4\eta \ge 2\xi-4\eta-1 > \eta,$$

where the first inequality comes from the fact that K is non-singleton and $\xi > 4\eta$, and the last inequality from the fact that $\xi > \frac{1+5\eta}{2}$. This again contradicts the optimality of $(\mathcal{C}^{\star},\chi^{\star})$, completing the entire proof of this case. ◀

The above lemma shows that each precluster $K \in \mathcal{K}$ is fully contained in some cluster $C^{\star} \in \mathcal{C}^{\star}$. We can also show that every non-singleton precluster must be colored the same as the cluster in $\mathcal{C}^{\star}$ containing it.

▶ **Lemma 18** (Restatement of Lemma 13 for CCC)**.** *For every cluster $C^{\star} \in \mathcal{C}^{\star}$ and any non-singleton precluster $K \in \mathcal{K}$ contained in $C^{\star}$, we have $\chi_{\mathsf{pre}}(K) = \chi^{\star}(C^{\star})$.*

Proof. Suppose toward contradiction that there exists a non-singleton K and $C_1^{\star}$ such that $K \subseteq C_1^{\star}$ but $\chi_{\mathsf{pre}}(K) \neq \chi^{\star}(C_1^{\star})$. We show that a solution $(\mathcal{C},\chi)$ obtained by splitting $C_1^{\star}$ into K of color $\chi_{\mathsf{pre}}(K)$ and $C_1^{\star}\setminus K$ of color $\chi^{\star}(C_1^{\star})$ has fewer disagreements than $(\mathcal{C}^{\star},\chi^{\star})$. The proof indeed follows from exactly the same argument as in the latter half of Case 2 in the proof of Lemma 17. ◀

▶ **Lemma 19.** *$(\mathcal{K},\chi_{\mathsf{pre}})$ is a constant-approximate solution.*

Proof. We show that the number of disagreements in $(\mathcal{K},\chi_{\mathsf{pre}})$ is within a constant factor of that in $(\mathcal{C}_{\mathsf{init}},\chi_{\mathsf{init}})$; since $(\mathcal{C}_{\mathsf{init}},\chi_{\mathsf{init}})$ is a constant-approximate solution, this completes the proof. To this end, we charge the edges newly disagreed in $(\mathcal{K},\chi_{\mathsf{pre}})$ due to the algorithm to the edges already disagreed in $(\mathcal{C}_{\mathsf{init}},\chi_{\mathsf{init}})$.

Consider any non-singleton cluster $C \in \mathcal{C}_{\mathsf{init}}$ of color $\ell := \chi_{\mathsf{init}}(C)$. Observe that, if a vertex $u \in C$ becomes a singleton precluster in $\mathcal{K}$, the set of newly disagreed edges is precisely $E^{\ell}(u,C)$. Let

$$B_1 := \{u \in C \mid |E^{-\ell}(u,C)| \ge \alpha(|C|-1)\} \text{ and}$$
$$B_2 := \{u \in C \mid |E^{+}(u,V\setminus C)| \ge \alpha(|C|-1)\}.$$

Note that $B_1 \cup B_2$ is the set of marked vertices in C right after iterating over V. We break the analysis into two cases depending on the size of $B_1 \cup B_2$.

Case 1. $|B_1 \cup B_2| < \beta(|C| - 1)$. In this case, the vertices in $C \setminus (B_1 \cup B_2)$ remain unmarked, and hence, it suffices to consider the vertices in $B_1 \cup B_2$. For any $u \in B_1$, observe that

$$\begin{aligned}|E^{\ell}(u, C)| &= |E(u, C)| - |E^{-\ell}(u, C)| = |C| - 1 - |E^{-\ell}(u, C)| \\ &\leq (1 - \alpha)(|C| - 1) \leq \frac{1 - \alpha}{\alpha}|E^{-\ell}(u, C)| < \frac{1}{\alpha}|E^{-\ell}(u, C)|,\end{aligned}$$

where the first and second inequalities are due to the definition of B_1; on the other hand, for $u \in B_2$, we have

$$|E^{\ell}(u, C)| \leq |C| - 1 \leq \frac{1}{\alpha}|E^{+}(u, V \setminus C)|,$$

where the last inequality is due to the definition of B_2. Hence, for any $u \in B_1 \cup B_2$, it is sufficient to cover $|E^{\ell}(u, C)|$ by charging $\frac{1}{\alpha}$ to each edge in $E^{-\ell}(u, C) \cup E^{+}(u, V \setminus C)$. Hence the total charged amount for the cluster C is at most

$$\frac{1}{\alpha}\left(2\,|\{uv \in E^{-\ell} \mid uv \subseteq C\}| + |\{uv \in E^{+} \mid |uv \cap C| = 1\}|\right).$$

Case 2. $|B_1 \cup B_2| \geq \beta(|C| - 1)$. Recall that, in this case, the algorithm marks all vertices in C. The total number of newly disagreed edges is at most $\binom{|C|}{2}$. Note that, by the definition of B_1,

$$|\{uv \in E^{-\ell} \mid uv \subseteq C\}| \geq \frac{1}{2}\sum_{u \in B_1}|E^{-\ell}(u, C)| \geq \frac{1}{2}\,|B_1| \cdot \alpha(|C| - 1),$$

and, by the definition of B_2,

$$|\{uv \in E^{+} \mid |uv \cap C| = 1\}| \geq \sum_{u \in B_2}|E^{+}(u, V \setminus C)| \geq |B_2| \cdot \alpha(|C| - 1).$$

We can thus derive that

$$\begin{aligned}&2\,|\{uv \in E^{-\ell} \mid uv \subseteq C\}| + |\{uv \in E^{+} \mid |uv \cap C| = 1\}| \\ &\geq \alpha(|B_1| + |B_2|)(|C| - 1) \\ &\geq \alpha\beta(|C| - 1)^2,\end{aligned}$$

where the last inequality is due to the condition that $|B_1| + |B_2| \geq |B_1 \cup B_2| \geq \beta(|C| - 1)$. This yields

$$\begin{aligned}\binom{|C|}{2} &= \frac{|C|(|C| - 1)}{2} \leq (|C| - 1)^2 \\ &\leq \frac{1}{\alpha\beta}\left(2\,|\{uv \in E^{-\ell} \mid uv \subseteq C\}| + |\{uv \in E^{+} \mid |uv \cap C| = 1\}|\right),\end{aligned}$$

where the first inequality is because C is non-singleton. It therefore suffices to cover the new disagreements due to C by charging $\frac{2}{\alpha\beta}$ to each edge in $\{uv \in E^{-\ell} \mid uv \subseteq C\}$ and $\frac{1}{\alpha\beta}$ to each edge in $\{uv \in E^{+} \mid |uv \cap C| = 1\}$, respectively.

Due to the above charging argument, we can see that every disagreement in $(\mathcal{C}_{\mathsf{init}}, \chi_{\mathsf{init}})$ is charged at most $\frac{2}{\alpha\beta}$ while sufficiently covering the cost incurred by the new disagreements in $(\mathcal{K}, \chi_{\mathsf{pre}})$. ◀

Hitting Axis-Parallel Segments with Weighted Points

Rajiv Raman
Indraprastha Institute of Information Technology Delhi, New Delhi, India

Siddhartha Sarkar
Indian Institute of Science, Bengaluru, India

Jatin Yadav
Indian Institute of Technology Delhi, New Delhi, India

Abstract

We study a geometric hitting-set problem in which the input consists of a set P of weighted points and a family $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ of axis-parallel segments in the plane. The goal is to select a minimum-weight subset of P that hits every segment in $\mathcal{S}$. Even restricted geometric hitting-set problems are known to be computationally hard, and for axis-parallel segments the standard decomposition into horizontal and vertical sub-instances yields only a simple factor-2 approximation.

We present an LP-rounding algorithm that breaks the factor-2 barrier. For the weighted problem, we obtain a randomized $(1 + 2/e)$-approximation by combining systematic rounding on horizontal lines with an exact repair step on residual vertical sub-instances. In the unweighted case, a sharper analysis gives a $(1 + 1/(e - 1))$-approximation. Finally, we consider the case where one of the sub-instances consists of lines instead of line segments, a problem considered by Fekete et al. (Geometric Hitting Set for Segments of Few Orientations, Theor. Comp. Sys., 62 (2) 2018),. In this case, we improve their result to obtain an approximation factor of $1 + 1/e$ and show that the problem is APX-hard. We also present algorithms for the generalization to d orientations, as well as PTASes for bounded-complexity subclasses of the unweighted Hitting Set problem.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Computational geometry; Theory of computation → Approximation algorithms analysis

Keywords and phrases Geometric Hitting Set, Approximation Algorithms, Computational Geometry, LP Rounding, Axis-Parallel Segments, PTAS, APX-hardness

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.14

Category APPROX

Related Version *Extended Version*: `https://arxiv.org/abs/2605.14499` [33]

1 Introduction

A *set system* $(P, \mathcal{S})$ consists of a set P of elements and a collection $\mathcal{S}$ of subsets of P. Given a set system, $(P, \mathcal{S})$, a set $P' \subseteq P$ is a *hitting set* if each set in $\mathcal{S}$ contains an element of P'. In the *unweighted* Hitting Set Problem, the goal is to find a hitting set of minimum cardinality, and in the *weighted* Hitting Set Problem, we are additionally given a weight function $w : P \to \mathbb{R}$ and the goal is to select a minimum weight hitting set P', i.e., a hitting set such that $\sum_{p \in P'} w(p)$ is minimized. The set systems considered in this paper are finite. We assume that $|P| = n$ and $|\mathcal{S}| = m$.

Without restrictions, the Hitting Set Problem is equivalent to the Set Cover Problem on the *dual set system*, where points in P and sets in $\mathcal{S}$ swap roles. For these general systems, $O(\log m)$-approximation algorithms exist with nearly matching lower bounds. However, we can often bypass these limits by exploiting the structure of restricted instances. This paper

© Rajiv Raman, Siddhartha Sarkar, and Jatin Yadav;
licensed under Creative Commons License CC-BY 4.0

focuses on *geometric set systems*, where elements and sets are defined by objects in the plane. While Hitting Set and Set Cover remain NP-hard in these settings, they frequently admit better approximation ratios than general systems. Despite this, a unified set of techniques and a complete characterization of these systems remain elusive.

There are two broad frameworks that yield improved approximation algorithms for Hitting Set and Set Cover. The first is via ϵ-nets and LP-rounding ([20, 8, 16, 36, 10]) that yield algorithms with approximation factors depending on the *combinatorial complexity* (such as *VC-dimension*, or *shallow-cell complexity*) of the instance. However, these techniques at best yield $O(1)$-approximation algorithms with large hidden constants even for simple objects such disks or squares.

A second framework is *local search*, that yields PTASes when the geometric set system is *non-piercing*, but works only in the unweighted setting [28, 13, 32]. Informally, a set $\mathcal{S}$ of objects in the plane is non-piercing if the boundary of each object in $\mathcal{S}$ is a simple Jordan curve, and for any $A, B \in \mathcal{S}$, $A \setminus B$ and $B \setminus A$ are both connected sets (see [32] for a precise definition). On the other hand, Chan and Grant [10] (see also Har-Peled and Quanrud [19]) showed even with nearly congruent *almost-squares*, the Set Cover and Hitting Set problems become APX-hard. Recently, Banik et al. [6] showed that the local search technique yields a PTAS for the Set Cover problem even when the geometric shapes are not non-piercing, i.e., they *pierce*, but in a bounded manner. However, no such result is known for the Hitting Set Problem.

Our goal is to develop improved approximation algorithms for structured, weighted geometric set systems. Specifically, we address the Hitting Set problem for axis-parallel line segments (formally defined below).

Weighted Geometric Hitting Set for Axis-Parallel Segments

Input: A finite set P of points in the plane with a non-negative weight function $w\colon P \to \mathbb{R}_{\geq 0}$, and a family $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ of axis-parallel segments (where $\mathcal{H}$ are horizontal and $\mathcal{V}$ are vertical).

Goal: Find a subset $Q \subseteq P$ that minimizes the total weight $\sum_{p \in Q} w(p)$ such that Q hits every segment in $\mathcal{S}$ (i.e., $Q \cap s \neq \emptyset$ for all $s \in \mathcal{S}$).

Axis-parallel segments generalize several tractable special cases. When all segments in $\mathcal{S}$ are parallel to a single axis, the problem becomes one-dimensional; alternatively, if segments only intersect those of orthogonal orientations, the problem maps to bipartite graphs. In both scenarios, the Hitting Set, Set Cover, and also the Independent Set problems are solvable in polynomial time because their natural LP-relaxations are integral, owing to totally unimodular constraint matrices [34]. Crucially, this total unimodularity breaks down in our setting, where segments possess both orientations and parallel segments overlap.

For Hitting Set, Set Cover, and Independent Set, a trivial 2-approximation follows from independently solving the one-dimensional sub-instances for each axis and combining the results. Remarkably, while this remains the best-known approximation factor for Set Cover and Independent Set, our primary contribution is breaking this factor-2 barrier for the weighted Hitting Set problem.

1.1 Our Contributions

Our main contribution is a suite of improved approximation algorithms for the Hitting Set problem on axis-parallel line segments and points in the plane, summarized in Table 1. For line segments in two orientations, we achieve a $(1 + 2/e) \approx 1.74$-approximation in the

weighted setting and a $(1 + 1/(e - 1)) \approx 1.58$-approximation in the unweighted setting. When one orientation consists of infinite lines instead of segments, we improve this to a $(1 + 1/e) \approx 1.36$-approximation. We further generalize our results to segments parallel to one of d directions. These results are obtained via a randomized rounding procedure on the natural linear program, but can be derandomized using the standard method of conditional expectations (See [33]). We also present a PTAS for specific restricted instances.

Table 1 The table above summarizes our results. Note that the previously best-known 2-approximation is considered a folklore result.

Problem	Approx. Factor	Complexity	Previous bounds
Wt. axis-parallel	$1 + 2/e$ (Th. 1)		2
Unwt. axis-parallel	$1 + 1/(e - 1)$ (Th. 2)		2, APX-hard [17]
d-dir wt.	$\ln d + \ln \ln d + O(1)$ (Th. 3)		d
Hor-seg, vert lines	$1 + 1/e$ (Thm. 5)	APX-hard (Thm. 6)	5/3 [17]
k-restricted unwt.	PTAS (Thm. 8)		NP-hard
Lines in 3-directions		APX-hard (Thm. 7)	4/3 [15], NP-hard [17]

▶ **Theorem 1** (Weighted). *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of horizontal and vertical line segments in the plane and let P be a set of points with a weight function $w : P \to \mathbb{R}_{\geq 0}$ such that each segment in $\mathcal{S}$ contains at least one point of P. Then, there is a randomized $(1 + 2/e)$-approximation algorithm for the weighted Hitting Set problem defined by $\mathcal{S}$ and P.*

For the unweighted setting, we obtain a better approximation factor.

▶ **Theorem 2** (Unweighted). *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of horizontal and vertical line segments in the plane and let P be a set of points such that each segment in $\mathcal{S}$ contains at least one point of P. Then, there is a randomized $(1 + 1/(e - 1))$-approximation algorithm for the Hitting Set problem defined by $\mathcal{S}$ and P.*

We can generalize the result for axis-parallel segments in the setting where each segment is parallel to one of d directions. Fekete et al. [17] studied the setting where each object to be hit is not a single segment, but is the union of h segments (each coming from one of d orientations). The authors provide an $(h \cdot d)$-approximation for this setting. Applying the result below, it is possible to obtain an improved $h \cdot (\log d + \log \log d + O(1))$-approximation.

▶ **Theorem 3** (d-Directions). *Let $\mathcal{S}$ be a set of segments, each parallel to one of d directions in the plane for some $d \geq 2$, and let P be a set of points with a weight function $w : P \to \mathbb{R}_{\geq 0}$ such that each segment in $\mathcal{S}$ contains at least one point of P. Then, there is a randomized algorithm that is a $(\ln d + \ln \ln d + O(1))$-approximation relative to the natural LP.*

We show that the natural LP-relaxation of the unweighted problem has an integrality gap of at least 5/4.

▶ **Theorem 4** (Integrality gap). *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of horizontal and vertical line segments in the plane and let P be a set of points such that each segment in $\mathcal{S}$ contains at least one point of P. The natural LP-relaxation of the problem has an integrality gap of at least 5/4.*

We also consider the setting when the segments in one of the orientations is replaced by lines, a variant considered by Fekete et al. [17], who gave a (5/3)-approximation for the unweighted case. We obtain a $1 + 1/e \approx 1.36$-approximation.

▶ **Theorem 5** (Lines and Segments). *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of horizontal line segments and vertical lines in the plane and let P be a set of points with a weight function $w : P \to \mathbb{R}_{\geq 0}$ such that each object in $\mathcal{S}$ contains at least one point of P. Then, there is a randomized $(1 + 1/e)$-approximation algorithm for the weighted Hitting Set problem defined by $\mathcal{S}$ and P.*

This setting with lines in one direction was shown to be NP-hard by Fekete et al. [17] if the point set consists of all points in the plane. However, if the set P is finite, we show that in fact, the problem is APX-hard.

▶ **Theorem 6** (APX-Hardness). *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of horizontal line segments and vertical lines in the plane and P be a finite set of points in the plane such that each object in $\mathcal{S}$ contains at least one point of P. The Hitting Set problem with $\mathcal{S}$ and P is APX-hard.*

If instead of line segments, we have lines in two dimensions, the Hitting Set problem with a set P of points is polynomial time solvable via a reduction to Edge Cover in bipartite graphs. Fekete [17] showed that if the set P consists of all points in the plane, then the problem is NP-hard. We sharpen their result to show that if we are additionally given a finite set P of points, then the problem becomes APX-hard.

▶ **Theorem 7** (3-Line-APX-hardness). *Let $\mathcal{S} = \mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z}$ be three sets of lines such that lines in the plane in each set $\mathcal{X}, \mathcal{Y}$ and $\mathcal{Z}$ are parallel. Let P be a set of points such that each line contains at least one point. The Hitting Set problem defined by $\mathcal{S}$ and P is APX-hard.*

Finally, we consider a restricted setting where each segment intersects segments of the opposite orientation that lie on at most k distinct parallel lines. We call this the k-restricted setting. In this setting, we show that the problem admits a PTAS.

▶ **Theorem 8** (k-restricted). *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of horizontal and vertical line segments in the plane such that each segment intersects segments of the opposite orientation lying on at most k parallel lines. Let P be a set of points such that each segment in $\mathcal{S}$ contains at least one point of P. Then, there is a PTAS for the Hitting Set problem with $\mathcal{S}$ and P.*

1.2 Related Work

ϵ-nets. Geometric Hitting Set and Set Cover approximations heavily depend on shape complexity. The ϵ-net framework[1] (see [26] for results on ϵ-nets) connects ϵ-net sizes to LP integrality gaps, where bounded VC-dimension yields $O(\log \mathrm{OPT})$-approximations [8, 16, 20]. Many geometric set systems have bounded VC-dimension, thus yielding an $O(\log \mathrm{OPT})$-approximation algorithms. Since the ϵ-net is obtained by uniform random sampling, the approximation algorithms extend to the weighted setting as well. While ϵ-nets of size $O(1/\epsilon)$ are known for some objects such as *pseudodisks*[2] and halfspaces [25, 31], translating to $O(1)$-approximation for Hitting Set, these smaller nets do not yield improved approximation algorithms for the weighted problem. If a geometric set system has linear *shallow-cell complexity*, then the *quasi-uniform sampling* technique [36, 12] yield improved approximation algorithms for the weighted problems. However, point-segment systems lack linear shallow-cell complexity.

[1] For a set system $(P, \mathcal{S})$, a set $P' \subseteq P$ is an ϵ-net if it is a hitting set for all sets $S \in \mathcal{S}$ such that $|S| \geq \epsilon |P|$.

[2] A set $\mathcal{S}$ of objects in the plane, each of whose boundary is a simple Jordan curve is a set of pseudodisks if for any $A, B \in \mathcal{S}$, the boundaries of A and B intersect in either 2 or 0 points.

Piercing and non-piercing. For the weighted setting, there is a QPTAS for the Set Cover problem [27] with non-piercing regions in the plane, but no such result is known for the Hitting Set problem. For the unweighted problem, local search provides PTASes for Hitting Set of pseudodisks, as shown by Mustafa and Ray [28]. The technique also extends to non-piercing regions [7, 32], and also yields PTASes for the Set Cover, Independent Set and Dominating Set problems. Recently, Banik et al. [6] showed that local search also yields PTASes for the Set Cover and Independent Set problems when the objects have a *bounded amount of piercing* [6]. Allowing objects to pierce in an unbounded manner however, renders the Hitting Set and Set Cover problems APX-hard, even for nearly congruent shapes like almost-square rectangles [10] and nearly equilateral triangles [19].

Segments. *Axis-Parallel Covering:* Fekete et al. [17] proved that the Hitting Set problem for axis-parallel segments is APX-hard. For the restricted case where one direction consists of lines, they established NP-hardness and provided a (5/3)-approximation. For Set Cover, Kowalska and Pilipczuk [22] showed APX-hardness for horizontal and vertical segments (even with δ-extensions) and W[1]-hardness for the exact weighted problem. They also developed several FPT algorithms and parameterized approximation schemes parameterized by the solution size k.

Independent Set: For the Independent Set (IS) problem, Marx [24] showed the axis-parallel case is in FPT, whereas arbitrary orientations are W[1]-hard. Parameterized approximation schemes also exist for the axis-parallel setting [14]. Furthermore, Caoduro et al. [9] established a $2-\epsilon$ lower bound on the ratio between clique cover and independence numbers for segment intersection graphs. Consequently, the natural LP-relaxation for IS on axis-parallel segments has an integrality gap of 2, contrasting with our Hitting Set results.

Arbitrary Orientations: For segments with arbitrary orientations, ϵ-nets have a known size of $O(1/\epsilon \log(1/\epsilon))$ because the VC-dimension is 2. However, Alon [2] and Balogh et al. [5] established an ϵ-net lower bound of $\Omega(1/\epsilon \log^{1/3}(1/\epsilon))$, limiting the best possible ϵ-net-based approximation to $O(\log^{1/3-o(1)} \mathrm{OPT})$. Finally, for the IS problem on arbitrary segments, there is a weighted QPTAS [1] and an unweighted $O(n^{\epsilon})$-approximation [18].

Rectangles. Axis-parallel segments can also be seen as a special case of axis-parallel rectangles. For the Hitting Set problem with axis-parallel rectangles, Aronov, Ezra and Sharir [3] obtained an $O(\log \log \mathrm{OPT})$-approximation by showing that this set system admits an ϵ-net of size $O(1/\epsilon \log \log 1/\epsilon)$, and this is the best approximation factor known. Pach and Tardos [29] showed that in fact, this bound is tight, and therefore, the natural LP-relaxation for the Hitting Set problem has an integrality gap of $\Omega(\log \log \mathrm{OPT})$.

Organization

Section 2 presents the main algorithm and analysis. Section 3 gives an explicit integrality-gap example. The extension to d-orientations is in Section 4. Section 5 records a separator-based PTAS for bounded-complexity subclasses. Section 6 and Section 7 give the hardness results, and we conclude in Section 8.

2 LP-rounding Algorithm

Let P be a set of n points in the plane with nonnegative weight function $w\colon P \to \mathbb{R}_{\geq 0}$. Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of m axis-parallel segments, where $\mathcal{H}$ is the set of horizontal segments and $\mathcal{V}$ is the set of vertical segments. For a segment s, let $P \cap s$ denote the candidate points

lying on s. We always assume $P \cap s \neq \emptyset$ for every $s \in \mathcal{S}$. When we refer to a horizontal or vertical line, we mean a line that contains at least one point of P. The natural LP relaxation for the Hitting Set Problem is

$$\begin{aligned} \min \quad & \sum_{p \in P} w(p)\, x_p \\ \text{s.t.} \quad & \sum_{p \in P \cap s} x_p \geq 1 \qquad \forall s \in \mathcal{S}, \\ & 0 \leq x_p \leq 1 \qquad \forall p \in P. \end{aligned}$$

Fix an optimal LP solution x, and let

$$\mathrm{LP} := \sum_{p \in P} w(p)\, x_p.$$

Since this is a relaxation of the hitting-set problem, $\mathrm{LP} \leq \mathrm{OPT}$.

We use one elementary fact repeatedly: if all candidate points and all segments lie on a single line, then the induced stabbing problem is an interval stabbing instance, whose constraint matrix has the consecutive-ones property and is therefore totally unimodular [34]. Consequently, the one-dimensional problem can be solved exactly in polynomial time.

Algorithm. The algorithm has two phases. In the first phase, we use randomized rounding to obtain a hitting set for the horizontal segments. In doing so, we ensure that each point p is in the hitting set produced with probability x_p, its LP-value. This solution partitions the vertical segments into *blocks*. Each block B is a one-dimensional problem, and the LP solution restricted to B is feasible for the intervals contained in B. Therefore, we can obtain a solution of value at most $\sum_{p \in B} w(p) x_p$. While the algorithm remains the same for the weighted and unweighted problems, the analysis for the unweighted problem can be somewhat sharpened.

We now describe Phase I, where we obtain a hitting set for the horizontal segments. Fix a horizontal line h and let $p_1, \ldots, p_k$ be the points on h from left to right. We define the cumulative sums a_i, $i = 0, \ldots, k$ as follows:

$$a_0 := 0, \qquad a_i := \sum_{j=1}^{i} x_{p_j} \quad (i = 1, \ldots, k),$$

For each i let $I_i := [a_{i-1}, a_i)$. Choose a random shift U_h uniformly from $[0, 1]$. We select point p_i in Phase I if and only if I_i contains an integer of the shifted lattice $U_h + \mathbb{Z}$. We perform this rounding independently for every horizontal line.

We claim that each point p is selected in Phase I with probability x_p, and that the points selected in Phase I constitute a hitting set for the horizontal segments.

▶ **Lemma 9.** *For each point $p \in P$, the probability that p is selected in Phase I is exactly x_p.*

Proof. If $p = p_i$, then the interval I_i has length $x_{p_i} \leq 1$. As the shift U_h varies uniformly over $[0, 1)$, the set of shifts for which I_i contains a point of $U_h + \mathbb{Z}$ has measure exactly $|I_i| = x_{p_i}$. Hence $\mathbb{P}[p_i \text{ is selected in Phase I}] = x_{p_i}$. ◀

▶ **Lemma 10.** *Every horizontal segment is hit by Phase I with probability* 1.

Proof. Let $s \in \mathcal{H}$ be a horizontal segment on line h, and suppose it contains the consecutive points $p_i, p_{i+1}, \ldots, p_j$. The union of the corresponding intervals is $[a_{i-1}, a_j)$, whose length is

$$a_j - a_{i-1} = \sum_{t=i}^{j} x_{p_t} \geq 1$$

by the LP constraint for s. Any interval of length at least 1 contains a point of $U_h + \mathbb{Z}$ for every shift $U_h \in [0, 1)$. Therefore at least one point of $s \cap P$ is selected. ◀

In Phase II, we solve the residual problem obtained on removing the points selected in Phase I and the segments in $\mathcal{S}$ hit by these points. This residual problem consists of a disjoint collection of vertical segments, each of which is called a *block*. Each block consists of a maximal consecutive run of unselected points on a vertical line and the segments that lie entirely in the block. Each block can be solved optimally either via dynamic programming, or by solving a natural LP-relaxation for the block. We now analyze the cost of the solution returned by the algorithm. First, we observe that the expected cost of the points selected in Phase I is exactly equal to the LP cost.

▶ **Lemma 11.** $\mathbb{E}[\text{cost}(\textit{Phase I})] = \text{LP}$

Proof. $\mathbb{E}[\text{cost(Phase I)}] = \sum_{p \in P} w(p)\mathbb{P}[p \text{ selected in Phase I}] = \sum_{p \in P} w(p)x_p = \text{LP}$, where the second-last equality follows from Lemma 9. ◀

We analyze Phase II separately for the weighted case, the unweighted case, and the case where vertical segments are replaced by lines.

2.1 Weighted Hitting Set

Consider a block B in Phase II. Let $x(B)$ denote the sum of the LP values of all the points contained in the block. If $x(B) < 1$, then observe that no segment is completely contained in B. We can therefore, safely ignore such blocks and only deal with blocks with $x(B) \geq 1$. Let the weight of a block B be $w(B) = \sum_{i \in B} w(i)x_i$. For a point $p \in P$, let A_p be the event that p is unselected in Phase I and the block containing p has total LP mass at least 1. Observe that an LP-relaxation for the Hitting Set problem restricted to a block B has an optimal solution at most $w(B)$ if its mass $x(B)$ is at least 1. Therefore,

$$\mathbb{E}[\text{cost(Phase 2)}] \leq \sum_{p \in P} w(p)x_p\,\mathbb{P}[A_p].$$

We now obtain an upper bound on $\mathbb{P}[A_p]$. Fix a vertical line ℓ with ordered points $1, 2, \ldots, m$ from bottom to top, and fix a point $p \in \{1, \ldots, m\}$. Define random variables D_p and U_p as follows.

- D_p is the total LP mass of the maximal consecutive run of unselected points ending at p. If p itself is selected in Phase I, then $D_p = 0$.
- U_p is the total LP mass of the maximal consecutive run of unselected points immediately above p.

If p is unselected, then $D_p + U_p$ is exactly the total LP mass of the block containing p. Therefore, $\mathbb{P}[A_p] \leq \mathbb{P}[D_p + U_p \geq 1]$. We now show that the probability that D_p or U_p has mass at least t decreases exponentially.

▶ **Lemma 12.** *For every $t \geq 0$, $\mathbb{P}[D_p \geq t] \leq e^{-t}$ and $\mathbb{P}[U_p \geq t] \leq e^{-t}$. Moreover, D_p and U_p are independent.*

Proof. We prove the claim for D_p; the proof for U_p is identical. Since the claim is trivially true for $t = 0$, we consider $t > 0$. If no suffix of points ending at p has total LP mass at least t, then $\mathbb{P}[D_p \geq t] = 0$. Otherwise, let $j \leq p$ be the largest index such that $\sum_{i=j}^{p} x_i \geq t$. If $D_p \geq t$, then all points $j, j+1, \ldots, p$ must be unselected in Phase I. Since Phase I decisions on a vertical line are independent, $\mathbb{P}[D_p \geq t] \leq \prod_{i=j}^{p}(1 - x_i) \leq \exp\Big(-\sum_{i=j}^{p} x_i\Big) \leq e^{-t}$.

Finally, D_p depends only on the decisions for points at or below p, while U_p depends only on the decisions strictly above p. These sets of decisions are independent, so D_p and U_p are independent. ◀

We require the following technical lemma on stochastic dominance (See [35], Theorem 1.A.3, or the full version of our paper [33] for a proof).

▶ **Lemma 13.** *Let X_1, X_2, Y_1, Y_2 be random variables such that X_1 is independent of X_2 and Y_1 is independent of Y_2. If $X_1 \preceq_{\text{st}} Y_1$ and $X_2 \preceq_{\text{st}} Y_2$, then $X_1 + X_2 \preceq_{\text{st}} Y_1 + Y_2$.*

Let $E_1, E_2 \sim \text{Exp}(1)$ be independent exponential random variables. The tail bound above implies $D_p \preceq_{\text{st}} E_1$ and $U_p \preceq_{\text{st}} E_2$. By Lemma 13, we have $D_p + U_p \preceq_{\text{st}} E_1 + E_2$. Therefore, $\mathbb{P}[D_p + U_p \geq 1] \leq \mathbb{P}[E_1 + E_2 \geq 1]$.. Since $E_1 + E_2$ has the $\Gamma(2, 1)$ distribution, $\mathbb{P}[E_1 + E_2 \geq 1] = e^{-1}(1 + 1) = 2/e$. Hence $\mathbb{P}[A_p] \leq 2/e$ for every point p, and therefore

$$\mathbb{E}[\text{cost(Phase 2)}] \leq \frac{2}{e} \sum_{p \in P} w(p) x_p = \frac{2}{e}\text{LP}.$$

▶ **Theorem 1** (Weighted). *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of horizontal and vertical line segments in the plane and let P be a set of points with a weight function $w : P \to \mathbb{R}_{\geq 0}$ such that each segment in $\mathcal{S}$ contains at least one point of P. Then, there is a randomized $(1 + 2/e)$-approximation algorithm for the weighted Hitting Set problem defined by $\mathcal{S}$ and P.*

Proof. Combining the bound on Phase I from Lemma 11, and the bound on A_p above, we obtain

$$\mathbb{E}[\text{cost(output)}] \leq \text{LP} + \frac{2}{e}\text{LP} = \left(1 + \frac{2}{e}\right)\text{LP} \leq \left(1 + \frac{2}{e}\right)\text{OPT}.$$ ◀

2.2 Unweighted Hitting Set

Fix a vertical line ℓ containing at least one vertical segment in $\mathcal{V}$. Let the points on this line be $q_1, \ldots, q_m$ ordered bottom to top. Let t denote the highest point such that the sum of the LP values of the points above t is at least 1. That is, $t = \max\Big\{j \in \{1, \ldots, m\} : \sum_{i=j}^{m} x_i \geq 1\Big\}$. Since the line contains at least one segment, the LP value on ℓ is ≥ 1. Therefore, we can assume that t is defined. In the unweighted setting, we can improve the analysis by observing that the value of an optimal solution for a block B is at most $\lfloor \sum_{p \in B} x_p \rfloor$.

For ease of analysis, we introduce a sentinel point 0 immediately below point 1, and declare it always selected. For every $i \in \{0, 1, \ldots, m\}$, let $B(i)$ denote the maximal block of consecutive unselected points immediately above i, and define $Y_i := \lfloor x(B(i)) \rfloor$, where $x(B(i)) := \sum_{p \in B(i)} x_p$. Every block on the line is either $B(0)$ or a block $B(i)$ following a selected point $i \geq 1$. Thus the Phase II cost on this line is at most $Y_0 + \sum_{i=1}^{m} \mathbf{1}[i \text{ selected}]\, Y_i$, where $\mathbf{1}[i \text{ selected}]$ is the indicator random variable that is 1 if i is selected, and 0 otherwise. Moreover, if $i \geq t$, then $\sum_{j=i+1}^{m} x_j < 1$, so $Y_i = 0$. Therefore only $Y_0, Y_1, \ldots, Y_{t-1}$ matter.

▶ **Lemma 14.** *For every $i \in \{0, 1, \ldots, t-1\}$, $\mathbb{E}[Y_i \mid i \text{ is selected}] \leq \frac{1}{e-1}$, where for $i = 0$ the conditioning is vacuous.*

Proof. Fix $i \in \{0, 1, \ldots, t-1\}$. For every integer $k \geq 1$,

$$\mathbb{P}\big[x(B(i)) \geq k \mid i \text{ is selected}\big] \leq e^{-k}.$$

Indeed, if the above probability is non-zero, let $r > i$ be the smallest index such that $\sum_{j=i+1}^{r} x_j \geq k$. Then all points $i+1, \ldots, r$ must not be selected in Phase I. Since the rounding decisions on a vertical line are independent and the event that i is selected depends only on point i, conditioning on the selection of i does not affect the points above it. Thus

$$\mathbb{P}\big[x(B(i)) \geq k \mid i \text{ is selected}\big] \leq \prod_{j=i+1}^{r} (1 - x_j) \leq \exp\left(-\sum_{j=i+1}^{r} x_j\right) \leq e^{-k}.$$

Using the tail-sum formula[3],

$$\begin{aligned}\mathbb{E}[Y_i \mid i \text{ is selected}] = \sum_{k \geq 1} \mathbb{P}\big[Y_i \geq k \mid i \text{ is selected}\big] &\leq \sum_{k \geq 1} \mathbb{P}\big[x(B(i)) \geq k \mid i \text{ is selected}\big] \\ &\leq \sum_{k \geq 1} e^{-k} = \frac{1}{e-1}.\end{aligned}$$

◀

▶ **Lemma 15.** *For every vertical line,* $\mathbb{E}[\text{cost}(\textit{Phase II on this line})] \leq \frac{1}{e-1} \sum_{i=1}^{m} x_i$.

Proof. Using the block decomposition above,

$$\mathbb{E}[\text{cost(Phase II on this line)}] \leq \mathbb{E}[Y_0] + \sum_{i=1}^{t-1} \mathbb{P}[i \text{ selected}]\, \mathbb{E}[Y_i \mid i \text{ selected}].$$

By Lemma 14, every conditional expectation is at most $1/(e-1)$, and point i is selected with probability x_i. Hence,

$$\mathbb{E}[\text{cost(Phase II on this line)}] \leq \frac{1}{e-1} + \sum_{i=1}^{t-1} \frac{x_i}{e-1} = \frac{1 + \sum_{i=1}^{t-1} x_i}{e-1}.$$

By the definition of t, we have $\sum_{i=t}^{m} x_i \geq 1$, so $1 + \sum_{i=1}^{t-1} x_i \leq \sum_{i=1}^{m} x_i$. Substituting this inequality completes the proof. ◀

▶ **Theorem 2** (Unweighted). *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of horizontal and vertical line segments in the plane and let P be a set of points such that each segment in $\mathcal{S}$ contains at least one point of P. Then, there is a randomized $(1 + 1/(e-1))$-approximation algorithm for the Hitting Set problem defined by $\mathcal{S}$ and P.*

Proof. Phase I still has expected cost exactly $\text{LP} = \sum_{p \in P} x_p$. Summing the previous line bound over all vertical lines gives $\mathbb{E}[\text{cost(Phase 2)}] \leq 1/(e-1)\text{LP}$. Therefore

$$\mathbb{E}[\text{cost(output)}] \leq \text{LP} + \frac{1}{e-1}\text{LP} = \left(1 + \frac{1}{e-1}\right)\text{LP} \leq \left(1 + \frac{1}{e-1}\right)\text{OPT}.$$

◀

The analysis above is asymptotically tight as we show below.

[3] If X is a non-negative integer-valued random variable ($X \in \{0, 1, 2, \ldots\}$), the expected value is: $\mathbb{E}[X] = \sum_{n=1}^{\infty} \mathbb{P}(X \geq n)$

2.3 Tightness of the Unweighted Analysis

We record a family of instances showing that the analysis in Section 2.2 is asymptotically tight. Let k tend to infinity, and set $n := k^2$. The point set is the $n \times n$ grid $P = [n] \times [n]$. The input contains every horizontal and vertical segment consisting of exactly k consecutive grid points.

First, $\mathrm{OPT} = n^2/k = k^3$. Indeed, the k disjoint length-k horizontal segments in each row show that every feasible solution has size at least $nk = k^3$. Conversely, the set

$$Q^\star := \{(i,j) \in [n] \times [n] : j - i \equiv 0 \pmod{k}\}$$

has exactly k points in every row and every column, with consecutive chosen points spaced exactly k apart. Hence every length-k horizontal or vertical segment contains one point of $Q^\star$, so $|Q^\star| = k^3 = n^2/k$.

The same disjoint-segment lower bound also applies to the LP, and the uniform assignment $x_p = 1/k$ for every grid point is feasible. Hence this assignment is an optimal LP solution. We now run the algorithm with this symmetric optimum. On each row, Phase I chooses exactly one residue class modulo k, independently and uniformly from row to row. Thus Phase I selects exactly k points on each row, and

$$\text{cost(Phase I)} = k^3 = \mathrm{OPT}.$$

It remains to compute the expected repair cost on the columns. Fix a column. A row is already selected in this column with probability $p := 1/k$, independently across rows. Let $q := 1 - p$ and let Z be the Phase II repair cost on this column. If a maximal run of unselected points has length L, then the residual vertical segments inside this run are exactly the length-k intervals contained in the run, and the optimum one-dimensional repair cost is $\lfloor L/k \rfloor$: the lower bound comes from the disjoint length-k intervals in the run, and selecting every kth point attains it.

For $a \geq 1$, let N_a be the number of unselected runs in the column whose length is at least ak. Then $Z = \sum_{a=1}^{k} N_a$. A run of length at least ak starts in the first row with probability q^{ak}; it starts at any later row with probability pq^{ak}, requiring the previous row to be selected and the next ak rows to be unselected. Therefore, with $\alpha := q^k = (1 - 1/k)^k$,

$$\mathbb{E}[Z] = \sum_{a=1}^{k} \left(q^{ak} + (n - ak)pq^{ak}\right) = \sum_{a=1}^{k} (k - a + 1)\alpha^a \geq k\frac{\alpha(1-\alpha^k)}{1-\alpha} - \frac{\alpha}{(1-\alpha)^2},$$

using $\sum_{a=1}^{k}(a-1)\alpha^a \leq \sum_{a=1}^{\infty} a \cdot \alpha^a = \alpha/(1-\alpha)^2$. Now, there are k^2 columns each contributing $\mathbb{E}[Z]$ to the expected cost of Phase II and $\mathrm{OPT} = k^3$. Hence, the ratio of the expected cost of Phase II and OPT satisfies:

$$\frac{\alpha(1-\alpha^k)}{1-\alpha} - \frac{1}{k}\frac{\alpha}{(1-\alpha)^2} \leq \frac{\mathbb{E}[\text{cost(Phase II)}]}{\mathrm{OPT}} \leq \frac{1}{e-1}, \tag{1}$$

where the second inequality follows from our analysis of the unweighted case in Section 2.2. Now, as $k \to \infty$, we have $\alpha = (1 - 1/k)^k \to e^{-1}$, thus the left hand side of (1) tends to $e^{-1}/(1 - e^{-1}) = 1/(e - 1)$. Thus, the approximation factor $(1 + 1/(e - 1))$ obtained in the unweighted analysis is asymptotically tight for the algorithm.

2.4 Horizontal Segments and Vertical Lines

In this section, we show that the approximation factor can be improved when in one of the directions, we have lines instead of line segments.

▶ **Theorem 5** (Lines and Segments). *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of horizontal line segments and vertical lines in the plane and let P be a set of points with a weight function $w : P \to \mathbb{R}_{\geq 0}$ such that each object in $\mathcal{S}$ contains at least one point of P. Then, there is a randomized $(1 + 1/e)$-approximation algorithm for the weighted Hitting Set problem defined by $\mathcal{S}$ and P.*

Proof. Each vertical line ℓ that is not hit by the Phase I points contributes $\min_{p\in\ell} w(p) \leq \sum_{p\in\ell} w(p)x_p$ to the Phase II cost, and each vertical line that is hit by the Phase I points contributes 0. Thus,

$$\mathbb{E}[\text{cost(Phase II)}] \leq \sum_{\ell\in\mathcal{V}} \mathbb{P}[\ell \text{ is not hit by Phase I}] \sum_{p\in\ell} w(p)x_p.$$

Now, since $\sum_{p\in\ell} x_p \geq 1$, we have that $\mathbb{P}[\ell \text{ is not hit by Phase I}] \leq \frac{1}{e}$, and hence the total expected cost of Phase II is upper bounded by $\frac{\text{LP}}{e}$. Thus, from Lemma 11, we get that the total expected cost is at most $(1 + 1/e)\text{LP}$. ◀

3 An Explicit Integrality-Gap Construction

In this section, we present an instance on which the integrality gap of the standard LP for the unweighted problem is 1.25.

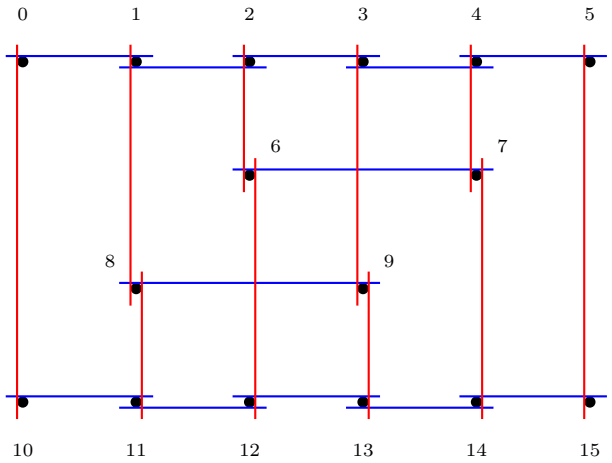


Figure 1 An integrality-gap instance for unweighted axis-parallel segment hitting. Segments are drawn with slight offsets for readability. The blue segments denote the horizontal segments $\mathcal{H}$, and the red segments denote the vertical segments $\mathcal{V}$.

Let $P = \{0, 1, \ldots, 15\}$ be the candidate points shown in Figure 1. Segments in the instance are formed by pairs of points that are on the same horizontal or vertical level, and have no other point on the segment connecting them. Thus, along the top chain we include every segment corresponding to consecutive pairs of points from $\{0, 1, 2, 3, 4, 5\}$, along the bottom chain every segment corresponding to consecutive pairs of points from $\{10, 11, 12, 13, 14, 15\}$, and so on.

▶ **Theorem 16.** *The instance of Figure 1 has integrality gap exactly 5/4 for the natural LP relaxation.*

Proof. Our hitting set instance is precisely an instance of vertex cover where the vertices are the points and for each segment, there is an edge between the vertices corresponding to the two points on the segment. Let G denote this graph. Set $x_p := 1/2$ for every point $p \in P$. Every segment in the construction contains at least two marked points, so this assignment is feasible. Hence the LP value is at most $\sum_{p\in P} x_p = 16 \cdot \frac{1}{2} = 8$.

In fact this bound is tight. The eight pairwise vertex-disjoint segments $\{0,1\}$, $\{2,3\}$, $\{4,5\}$, $\{6,7\}$, $\{8,9\}$, $\{10,11\}$, $\{12,13\}$, $\{14,15\}$ already force $\sum_{p\in P} x_p \geq 8$, so LP $= 8$. It remains to compute the optimum integral value. Since this is a vertex-cover instance,

$$\mathrm{OPT} = |V(G)| - \alpha(G) = 16 - \alpha(G),$$

where $\alpha(G)$ is the maximum independent-set size. The set $\{0,2,4,8,12,14\}$ is independent, so $\alpha(G) \geq 6$. For the reverse inequality, consider the 6-cycle $C = \{2,3,9,13,12,6\}$ and the two 5-cycles $L = \{0,1,8,11,10\}$ and $R = \{4,5,15,14,7\}$. Every independent set contains at most 3 vertices from C and at most 2 vertices from each of L and R. If an independent set I has $|I \cap C| \leq 2$, then immediately $|I| \leq 2+2+2 = 6$.

Suppose instead that $|I \cap C| = 3$. Then $I \cap C$ must be one of the two alternating triples $\{2,9,12\}$ or $\{3,6,13\}$. If I contains $\{2,9,12\}$, then the edges $(1,2)$, $(8,9)$, and $(11,12)$ exclude $1,8,11$ from the left 5-cycle. Only 0 and 10 remain available in L, and they are adjacent, so $|I \cap L| \leq 1$. Since $|I \cap R| \leq 2$, we get $|I| \leq 3+1+2 = 6$. The case of the other alternating triple ($\{3,6,13\}$) is symmetric.

Therefore $\alpha(G) = 6$, and hence OPT $= 16 - 6 = 10$. The integrality gap is

$$\frac{\mathrm{OPT}}{\mathrm{LP}} = \frac{10}{8} = \frac{5}{4}.$$

◀

4 Generalization to d orientations (weighted case)

Suppose segments are allowed to have d distinct orientations $\theta_1, \theta_2, \ldots, \theta_d$, and let $S^{(r)}$ denote the set of segments of orientation θ_r. The LP relaxation is unchanged:

$$\min \sum_{p\in P} w(p)x_p \qquad \text{s.t.} \sum_{p\in P\cap s} x_p \geq 1 \ \forall s \in \bigcup_{r=1}^{d} S^{(r)}, \quad 0 \leq x_p \leq 1.$$

Let x be an optimal LP solution with value LP.

Algorithm with a parameter k. Fix $k \in \{1,2,\ldots,d\}$ and call $\theta_1,\ldots,\theta_k$ the primary orientations. For every line of each primary orientation, independently run the same random-start systematic rounding used above, after rotating coordinates along that line. The output of the primary step is the union of all points selected by these k rounds. Therefore its expected cost is at most

$$\sum_{p\in P} w(p)\mathbb{P}[p \text{ is selected in at least one primary round}] \leq \sum_{p\in P} w(p) \sum_{t=1}^{k} x_p = k \cdot \mathrm{LP}.$$

All segments in the primary orientations are hit with probability one.

For every remaining orientation $r > k$, we then repair the residual one-dimensional instances on lines of orientation θ_r. These repairs may be performed independently for all remaining orientations and then unioned with the primary solution. This can only overcount the final cost, so it is enough to bound the sum of the repair costs orientation by orientation.

▶ **Lemma 17.** *Fix a remaining orientation θ_r and a point p on a line of orientation θ_r. Let A_p be the event that p survives the primary step and lies in a surviving block of LP mass at least 1 along that line. Then $\mathbb{P}[A_p] \leq (k+1)e^{-k}$.*

Proof. Fix a line ℓ of orientation θ_r and order its points along ℓ. In one primary orientation, a point i survives with probability $1 - x_i$; hence after all k primary rounds it survives with probability $(1 - x_i)^k$. Moreover, for any primary orientation θ_t and any two distinct points on ℓ, the two θ_t-lines through these points are distinct; otherwise a θ_t-line would meet ℓ in two points, forcing it to be ℓ, contrary to $\theta_t \neq \theta_r$. Thus the collections of primary-line shifts determining the survival of distinct points on ℓ are disjoint and independent, so the survival events of distinct points on ℓ are independent.

Now repeat the weighted-case block analysis from the proof of Lemma 12 with "survives the primary step" in place of "is unselected in Phase I". Let D_p be the LP mass of the maximal surviving run ending at p, and let U_p be the LP mass of the maximal surviving run immediately after p. If A_p occurs, then $D_p + U_p \geq 1$.

For any $t \geq 0$, the same suffix argument as in the proof of Lemma 12 gives

$$\mathbb{P}[D_p \geq t] \leq \prod_{i \in J}(1 - x_i)^k \leq \exp\left(-k \sum_{i \in J} x_i\right) \leq e^{-kt},$$

where J is a minimal consecutive suffix ending at p with LP mass at least t; if no such suffix exists, the probability is 0. Similarly, $\mathbb{P}[U_p \geq t] \leq e^{-kt}$, and D_p, U_p are independent. Thus D_p and U_p are stochastically dominated by independent exponential random variables E_1, E_2 of rate k. Therefore,

$$\mathbb{P}[A_p] \leq \mathbb{P}[D_p + U_p \geq 1] \leq \mathbb{P}[E_1 + E_2 \geq 1] = (k + 1)e^{-k},$$

where the second inequality follows from Lemma 13. ◀

Thus, we get that for each remaining orientation $r > k$,

$$\mathbb{E}[\text{repair cost for orientation } \theta_r] \leq \sum_{p \in P} x_p w(p) \mathbb{P}[A_p] \leq (k + 1)e^{-k}\text{LP}.$$

Summing over the $d - k$ remaining orientations and adding the primary cost yields

$$\mathbb{E}[\text{total cost}] \leq \left(k + (d - k)(k + 1)e^{-k}\right)\text{LP}. \tag{2}$$

▶ **Theorem 3** (d-Directions). *Let $\mathcal{S}$ be a set of segments, each parallel to one of d directions in the plane for some $d \geq 2$, and let P be a set of points with a weight function $w : P \to \mathbb{R}_{\geq 0}$ such that each segment in $\mathcal{S}$ contains at least one point of P. Then, there is a randomized algorithm that is a $(\ln d + \ln \ln d + O(1))$-approximation relative to the natural LP.*

Proof. Take $k = \min(d, \lceil \ln d + \ln \ln d \rceil)$. For large enough d, k equals $\lceil \ln d + \ln \ln d \rceil$, and thus $(d - k)(k + 1)e^{-k} = O(1)$, yielding the claimed approximation ratio when substituted in (2). ◀

We can further generalize this result to the case when each object to be hit is a union of up to h segments, each of which has one of d orientations.

▶ **Corollary 18.** *There exists a randomized algorithm that, given an instance of the weighted Hitting Set problem where each object is a union of at most h segments of d distinct orientations, computes a solution with an expected approximation factor of $h \cdot (\ln d + \ln \ln d + O(1))$ relative to the natural LP.*

Proof. Let x be a feasible solution to the natural LP. Since each object O is a union of at most h segments, the LP constraint $\sum_{p\in O} x_p \geq 1$ implies there is some segment $s \subseteq O$ such that $\sum_{p\in s} x_p \geq 1/h$. Choose one such segment for each object, and define $y_p := \min\{1, hx_p\}$. Then $0 \leq y_p \leq 1$, $\sum_p w(p)y_p \leq h \cdot \mathrm{LP}$, and for every chosen segment s we have $\sum_{p\in s} y_p \geq 1$: if some $p \in s$ has $hx_p \geq 1$ this is immediate, and otherwise $\sum_{p\in s} y_p = h\sum_{p\in s} x_p \geq 1$. Thus y is feasible for the chosen segment instance, and hitting these chosen segments hits every original object. Running the algorithm for d orientations on y yields the claimed expected approximation factor. ◀

5 Local Search PTASes for k-restricted instances

In this section, we show that if we restrict the input so that each segment intersects segments of the opposite orientation lying on at most k-distinct lines, then we obtain a PTAS for the unweighted Hitting Set problem. The results of Raman and Ray [32] only work if the objects are non-piercing. While Banik et al. [6] works for the case where each object has a bounded amount of piercing, their results only work for the Set Cover and Independent Set problems. The restriction on the number of such intersections is essential, as otherwise, as Section 6 shows, the problem is APX-hard.

▶ **Definition 19.** *A family $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ of axis-parallel segments is* k-restricted *if every segment in $\mathcal{H}$ intersects segments of $\mathcal{V}$ lying on at most k parallel lines, and every segment in $\mathcal{V}$ intersects segments of $\mathcal{H}$ lying on at most k parallel lines.*

Observe that in a k-restricted instance, a segment can intersect an unbounded number of segments of the opposite orientation. The only restriction is that these segments lie on at most k distinct parallel lines.

The local-search algorithm is standard. Fix an integer parameter t. Starting from any feasible hitting set, repeatedly replace at most t objects of the current solution by a strictly smaller feasible set, as long as such an improving swap exists. For fixed t, this process terminates in polynomial time. The key issue is to prove that a t-locally optimal solution is globally near-optimal when $t = t(\varepsilon)$ is chosen large enough [32, 4, 13, 28].

The analysis relies on showing the existence of a suitable *exchange graph* $G = (L \cup O, E)$ on a locally optimal solution L and an optimal solution O. This exchange graph has two properties: the *local property* that for any $L' \subseteq L$, $(L' \setminus L) \cup N_G(L')$ is also a hitting set for $\mathcal{S}$, where $N_G(L')$ are the neighbors of L' in the exchange graph G, and the *separator property* that G comes from a hereditary family $\mathcal{G}$ of graphs with sub-linear size separators. That is, there is an absolute constant $\epsilon > 0$ such that any graph $H \in \mathcal{G}$ has a separator of size $O(|V(H)|^{1-\epsilon})$[4].

Given two disjoint hitting sets R and B for a set $\mathcal{S}$ of axis-parallel segments, we show the existence of a suitable exchange graph. A PTAS then follows by standard arguments.

▶ **Lemma 20.** *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of k-restricted horizontal and vertical segments and let R and B be two disjoint hitting sets for the segments $\mathcal{S}$. Then, there is a local-exchange graph $G = (R \cup B, E)$ that satisfies the local exchange and the separator properties.*

[4] A balanced separator, or just a separator for a graph $G = (V, E)$ is a subset $S \subseteq V$ of vertices whose removal partitions the graph such that no resulting connected component contains more than $\alpha|V|$ vertices, where $\alpha < 1$ is a fixed constant, typically $1/2$ or $2/3$. A graph admits a sub-linear separator if there exists an absolute constant $\epsilon > 0$ such that $|S| = O(|V|^{1-\epsilon})$.

Proof. The graph $G = (R \cup B, E)$ is constructed as follows. Starting with the empty graph on $R \cup B$, for each segment $s \in \mathcal{S}$, let R_s and B_s denote respectively, the points of R in s and the points of B in s. Suppose no point in R_s is adjacent to a point in B_s, we choose an arbitrary pair of points, one in R_s and the other in B_s that are consecutive along s and add the edge $\{r, b\}$ to G as a straight-line segment along s. We claim that G is the desired local-exchange graph. We say that the points r and s are *responsible* for hitting s.

Let $R' \subseteq R$. We claim that $T = (R \setminus R') \cup N_G(R')$ is a hitting set. Indeed, for a segment s, if the vertex r responsible for hitting s is not selected in R', then s is still hit by a point in the set T. If not, since the vertex $b \in B$ responsible for hitting s is adjacent to r, it implies that s is hit by b. Therefore, G satisfies the local-exchange property.

Consider the natural embedding of G in the plane, where the edges are drawn as straight-line segments between the points. By construction, each edge in G is only intersected by edges of the opposite orientation. Since any edge lies along a segment, and the instance is k-restricted, it implies that each edge of G is crossed by at most k other edges. That is G is a *k-planar graph*[5]. k-planar graphs form a hereditary family as removing vertices leaves the induced graph k-planar.

Since k-planar graphs have $O(\sqrt{k}n)$ edges [30], where $n = |R| + |B|$ and any two segments intersect in at most one point we obtain a plane graph H with $O(k^{3/2}n)$ vertices, by splitting each pair of crossing edges by adding a new *splitting vertex* at their intersection point. We assign a weight of 1 to each vertex of G and a weight of 0 to each splitting vertex. By the Planar Separator Theorem [23], there is a balanced separator S of H of size $O(\sqrt{|V(H)|}$ such that both sides of the partition has weight at most 2/3 the total weight of the vertices in H. To obtain a separator in G, for each vertex in S, if a splitting vertex s lies in S, we add the four end-points corresponding to the pair of crossing edges defined by s. The resulting set S' is a balanced separator in G, and has size $O(|S|)$. Therefore, we obtain a sub-linear sized separator in G. ◀

▶ **Theorem 8** (k-restricted). *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of horizontal and vertical line segments in the plane such that each segment intersects segments of the opposite orientation lying on at most k parallel lines. Let P be a set of points such that each segment in $\mathcal{S}$ contains at least one point of P. Then, there is a PTAS for the Hitting Set problem with $\mathcal{S}$ and P.*

Proof sketch. Let L be a t-locally optimal Hitting Set and let O be an optimal Hitting Set. We can assume that $L \cap O = \emptyset$, as otherwise, we can remove the points in $L \cap O$ and the segments hit by them, and consider the residual instance. This can only improve the approximation factor. By Lemma 20, there is an exchange graph satisfying the local-exchange property. Now, a PTAS follows directly via the techniques in [28]. ◀

6 APX-hardness

The proof of the following theorem is based on the proof of APX-hardness of Minimum Hitting Set for orthogonal segments by Fekete et al. [17]. Fekete et al. show NP-hardness for the *piercing problem*, i.e., when we are allowed to choose any point in $\mathbb{R}^2$ to hit the segments. On the other hand, we show that if P is restricted to a discrete set, the problem becomes APX-hard.

[5] A graph is k-planar if it has an embedding in the plane such that each edge is crossed by at most k other edges

▶ **Theorem 6** (APX-Hardness). *Let $\mathcal{S} = \mathcal{H} \cup \mathcal{V}$ be a set of horizontal line segments and vertical lines in the plane and P be a finite set of points in the plane such that each object in $\mathcal{S}$ contains at least one point of P. The Hitting Set problem with $\mathcal{S}$ and P is APX-hard.*

Proof. We give an approximation-preserving reduction from Max-2SAT(3), the maximum satisfiability problem for CNF formulae with two literals per clause, in which each variable appears in at most three clauses. This problem is APX-hard [21].

Let Φ be an instance of Max-2SAT(3) with variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$. We may assume that every variable appears in at least one clause; isolated variables can be deleted. Since each clause contains two literals, the total number of literal occurrences is $2m$; therefore, $n \leq 2m$.

Let OPT denote the maximum number of clauses of Φ that can be satisfied. Also, since a uniformly random truth assignment satisfies each 2-literal-clause with probability at least $1/2$, we have $\text{OPT} \geq \frac{m}{2}$.

We construct an instance of Minimum Hitting Set as follows. For each variable x_i, let $d_i \in \{1, 2, 3\}$ be the number of occurrences of x_i in Φ. We construct a variable gadget G_i consisting of $2d_i + 4$ geometric objects: $2d_i + 2$ horizontal segments and two vertical lines. For each variable x_i, there is vertical line L_i at the left end and there is a vertical line R_i at the right end. These objects are arranged so that their intersection graph is an even cycle of length $2d_i + 4$, and no point intersects three objects of the variable gadget. Equivalently, the only points that hit two objects of G_i are the consecutive intersections along this cycle. Refer to Figure 2.

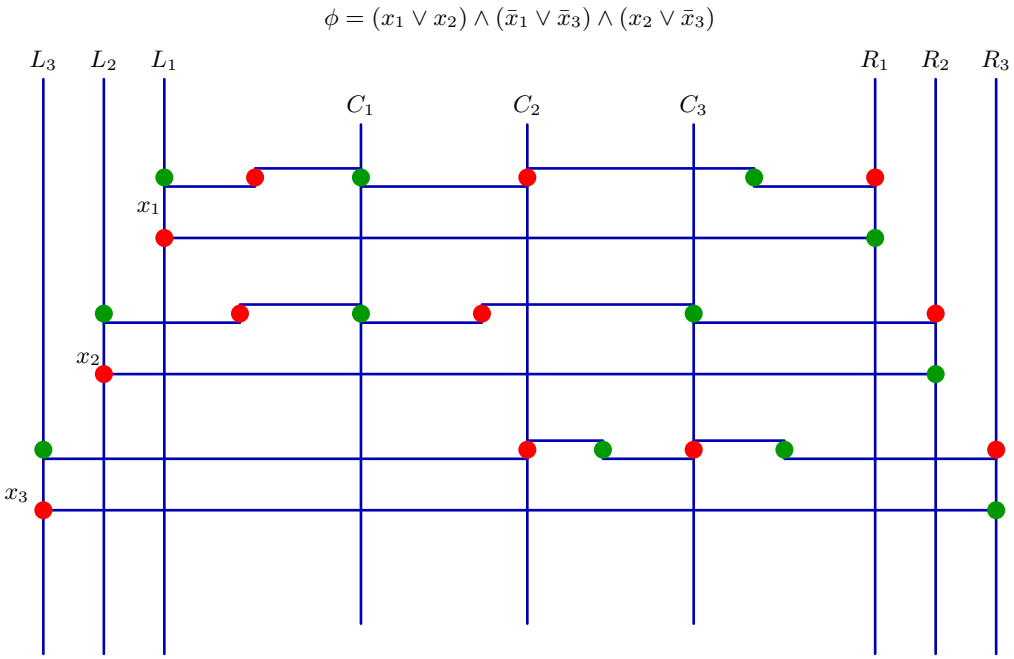


Figure 2 A hitting-set instance corresponding to $\phi = (x_1 \vee x_2) \wedge (\bar{x}_1 \vee \bar{x}_3) \wedge (x_2 \vee \bar{x}_3)$.

Color the consecutive intersection points of this cycle alternately red and green. Since G_i has $2d_i + 4$ objects and every point hits at most two objects of G_i, any hitting set for G_i has size at least $b_i := d_i + 2$.

Moreover, in a hitting set of size exactly b_i, every selected point must lie at an intersection of two consecutive objects of G_i. Therefore, it corresponds to a perfect matching of the cycle. Thus there are exactly two canonical optimum hitting sets for G_i: the set of all red points and the set of all green points. We interpret the green solution as setting $x_i = \text{true}$ and the red solution as setting $x_i = \text{false}$.

For each clause C_j, introduce a vertical line ℓ_j. If the literal x_i occurs in C_j, we make ℓ_j pass through the green point of G_i reserved for this occurrence. If the literal $\overline{x_i}$ occurs in C_j, we make ℓ_j pass through the corresponding red point of G_i. The lines and input points are placed such that these are the only points at which a clause line passes through. Thus, each clause line passes through exactly two input points. Let $B := \sum_{i=1}^{n} b_i$. Since $\sum_i d_i = 2m$, we get $B = 6m$.

We first relate satisfying assignments to hitting sets. Given an optimal truth assignment satisfying OPT clauses, choose in each variable gadget the canonical red or green hitting set corresponding to the truth value of the variable. This uses exactly B points. Every satisfied clause line is hit at the corresponding literal point. Each unsatisfied clause line can be hit by one additional point placed anywhere on that line. Therefore, if k denotes the optimum solution size of the constructed hitting-set instance, then $k \leq B + (m - \text{OPT})$. In particular, $k \leq B + m \leq 7m$.

Conversely, let H be any hitting set for the constructed instance. We normalize H without increasing its cardinality. Consider a variable gadget G_i. Since G_i has $2b_i$ objects and no point hits more than two objects of G_i, the points of H used to hit objects of G_i have cardinality at least b_i.

If the selected points of H for the variable gadget G_i are consistent, that is, all are red or all are green, then we replace the points of H used inside G_i by the corresponding canonical optimum set of b_i red or green points. This does not increase the number of points.

If the selected literal points are inconsistent, then H uses literal points of both colors. Since the two canonical solutions are the only size-b_i hitting sets for G_i, the portion of H used for G_i has size at least $b_i + 1$. We replace it by one canonical optimum set of size b_i, choosing the majority color among the selected literal points. Since $d_i \leq 3$, at most one clause line that was previously hit through a minority-colored literal point becomes unhit. We use the one saved point to hit that clause line separately. Again, the total number of points does not increase.

Applying this operation independently to all variable gadgets, we obtain a hitting set of the form $H_v \cup H_c$, with $|H_v| = B$ and $|H_v| + |H_c| \leq |H|$.

The set H_v is canonical in every variable gadget, and hence defines a truth assignment. The set H_c consists of additional points placed on clause lines. Every clause not satisfied by the assignment induced by H_v must be hit by a point of H_c. Therefore, if alg is the number of clauses satisfied by the induced assignment, then

$$\text{alg} \geq m - |H_c|.$$

Now suppose there is a polynomial-time $(1+\varepsilon)$-approximation algorithm for the hitting-set problem on horizontal segments and vertical lines. Applied to the constructed instance, it returns a hitting set H with $|H| \leq (1+\varepsilon)k$, and after the normalization above, $|H_c| \leq |H| - B \leq (1+\varepsilon)k - B$. Hence,

$$\text{alg} \geq m - |H_c| \quad \geq m - (1+\varepsilon)k + B = (m + B - k) - \varepsilon k.$$

Since $k \leq B + (m - \text{OPT})$, we have $m + B - k \geq \text{OPT}$, Thus, $\text{alg} \geq \text{OPT} - \varepsilon k$. Using $k \leq 7m$ and $\text{OPT} \geq m/2$, we obtain

$$\varepsilon k \leq 7\varepsilon m \leq 14\text{OPT}.$$

Therefore,

$$\text{alg} \geq (1 - 14\varepsilon)\,\text{OPT}.$$

Consequently, a PTAS for minimum hitting set with horizontal segments and vertical lines would imply a PTAS for Max-2SAT(3), contradicting its APX-hardness. Hence the hitting-set problem for horizontal segments and vertical lines is APX-hard. ◀

7 Discrete 3-Slope Line Cover is APX-hard

Discrete 3-Slope-Line-Cover (3SLC) refers to the problem of computing a minimum-cardinality hitting set for a set $\mathcal{S}$ of lines having three distinct slopes with a subset of a given set P of points. In this section, we prove that this problem is APX-hard [17].

▶ **Definition 21.** *(L-reduction) [11] A pair of functions f and g, both computable in polynomial time, is an L-reduction from an optimization problem $\mathcal{P}$ to an optimization problem $\mathcal{P}'$ if there are positive constants α and β such that for each instance x of $\mathcal{P}$, the following hold:*

- *(L1) The function f maps instances of $\mathcal{P}$ to instances of $\mathcal{P}'$ such that $OPT(f(x)) \leq \alpha \cdot OPT(x)$.*
- *(L2) The function g maps feasible solutions y of $f(x)$ to feasible solutions $g(y)$ of x such that $|c_x(g(y)) - OPT(x)| \leq \beta.|c_{f(x)}(y) - OPT(f(x))|$, where c_x and $c_{f(x)}$ are the cost functions of the instances x and $f(x)$ respectively.*

▶ **Theorem 7** (3-Line-APX-hardness). *Let $\mathcal{S} = \mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z}$ be three sets of lines such that lines in the plane in each set $\mathcal{X}, \mathcal{Y}$ and $\mathcal{Z}$ are parallel. Let P be a set of points such that each line contains at least one point. The Hitting Set problem defined by $\mathcal{S}$ and P is APX-hard.*

Proof. We give an L-reduction from Special-3SC, which is APX-hard [11]. Recall that an instance of Special-3SC is obtained from a cubic graph $G = (V, E)$. For every vertex $v_t \in V$, there are four elements w_t, x_t, y_t, z_t, and for every edge $e_i \in E$, there is an element a_i. If the three edges incident to v_t are e_i, e_j, e_k, with $i < j < k$, then the instance contains the five sets $\{w_t, a_i\}, \{w_t, x_t\}, \{x_t, y_t, a_j\}, \{y_t, z_t\}, \{z_t, a_k\}$. Moreover, every element a_i belongs to exactly two sets.

We first construct an equivalent finite discrete 3SLC instance in the dual plane. Let L_1, L_2, L_3 be three vertical lines, from left to right. We place the elements w_t, y_t on L_1, the elements x_t, z_t on L_2, and the elements a_i on L_3. The placement is chosen so that, for each v_t with incident edges e_i, e_j, e_k, $i < j < k$, there are five non-vertical candidate lines whose intersections with the constructed point set are exactly

$$\{w_t, a_i\}, \quad \{w_t, x_t\}, \quad \{x_t, y_t, a_j\}, \quad \{y_t, z_t\}, \quad \{z_t, a_k\}.$$

We briefly justify that such a placement can be found in polynomial time. Put the edge points $a_1, \ldots, a_n$ on L_3 in this order. For each vertex v_t, reserve two small open intervals $I_t^1 \subset L_1$ and $I_t^2 \subset L_2$, with these intervals pairwise disjoint over different vertices, such that every line joining a point of I_t^1 to an edge point a_r meets L_2 inside I_t^2. Such intervals are obtained, for example, by taking disjoint thin vertical cross-sections of triangles with common base on L_3.

Now choose $y_t \in I_t^1$ and let x_t be the intersection of $\overline{y_t a_j}$ with L_2. Choose $w_t \in I_t^1$ and $z_t \in I_t^2$. At each step, the forbidden choices are only those causing one of the five intended candidate lines to pass through an unintended point. There are finitely many such forbidden positions. Hence we can choose rational points avoiding all of them. See Figure 3.

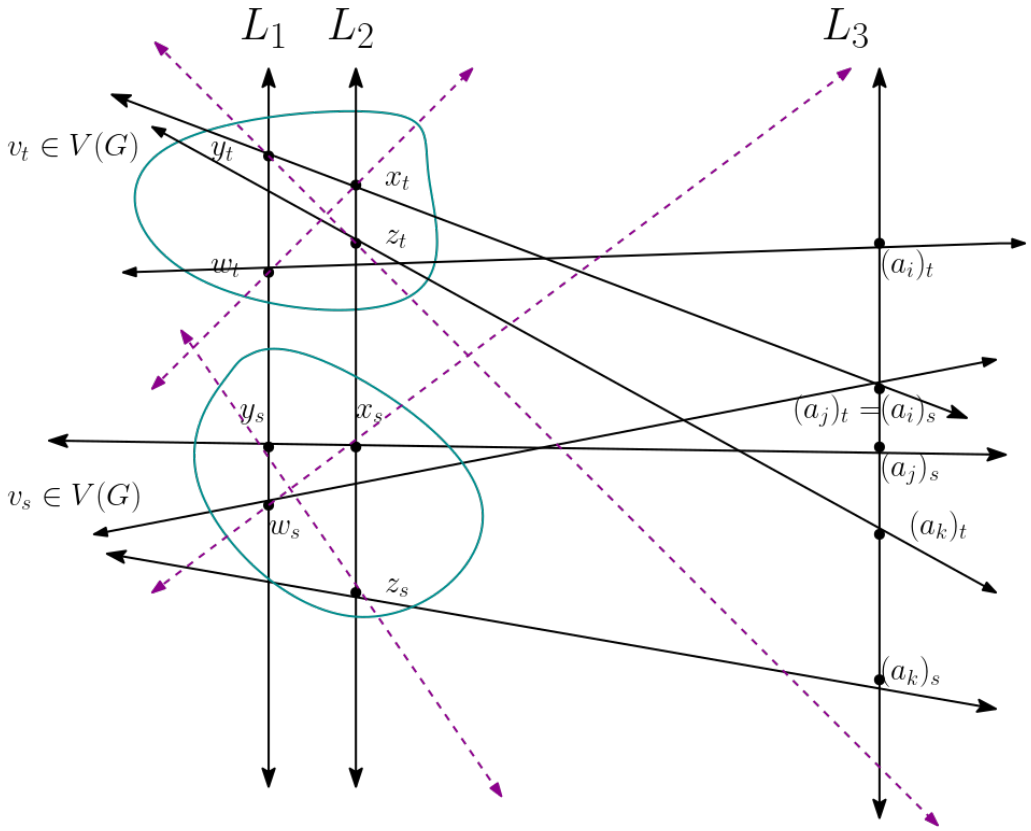


Figure 3 Two vertex gadgets for $v_s, v_t \in V$ in the dual construction, where $(v_s, v_t) \in E$. For each vertex v_t incident to edges e_i, e_j, e_k, $i < j < k$, the points w_t, y_t lie on L_1 and x_t, z_t lie on L_2. The three solid lines encode the sets $\{w_t, a_i\}$, $\{x_t, y_t, a_j\}$, and $\{z_t, a_k\}$; the dashed magenta lines encode the non-edge sets $\{w_t, x_t\}$ and $\{y_t, z_t\}$.

It follows that the constructed dual finite line-cover instance has exactly the same set system as the Special-3SC instance: selecting a candidate line is equivalent to selecting the corresponding set, and a subfamily covers all points if and only if the corresponding sets cover the universe. Therefore

$$\mathrm{OPT}_{\mathrm{dual}} = \mathrm{OPT}_{\mathrm{SC}},$$

where $\mathrm{OPT}_{\mathrm{dual}}$ denotes the optimal solution value for the dual line-cover instance and $\mathrm{OPT}_{\mathrm{SC}}$ denotes the optimal solution value for the Special-3SC instance.

It remains to pass from this dual finite line-cover instance to Discrete 3-Slope-Line-Cover. We use the standard point-line duality

$$p = (p_x, p_y) \longleftrightarrow p^* : b = p_x a - p_y,$$

and

$$\ell : y = ax - b \longleftrightarrow \ell^* = (a, b),$$

which preserves incidences. The three vertical lines L_1, L_2, L_3 in the dual plane become three slopes in the primal plane. The dual points become the input lines S, and the dual candidate lines become the finite candidate point set P. Hence we obtain an instance of Discrete

3-Slope Line Cover. Since incidence and cardinality are preserved, feasible covers in the dual line-cover instance are in one-to-one correspondence with feasible hitting sets in the resulting Discrete 3-Slope-Line-Cover instance.

Thus, for the reduction f,

$$\mathrm{OPT}_{\mathrm{3SLC}}(f(I)) = \mathrm{OPT}_{\mathrm{SC}}(I),$$

so condition (L1) of an L-reduction holds with $\alpha = 1$. Given any feasible solution to $f(I)$, mapping each selected point back by duality gives the corresponding selected set of the Special-3SC instance, with the same cardinality. Therefore

$$|c_{\mathrm{SC}}(g(Y)) - \mathrm{OPT}_{\mathrm{SC}}(I)| = |c_{\mathrm{3SLC}}(Y) - \mathrm{OPT}_{\mathrm{3SLC}}(f(I))|\,,$$

so condition (L2) holds with $\beta = 1$. The construction is polynomial-time computable. Hence Discrete 3-Slope-Line-Cover is APX-hard. ◀

8 Conclusion

We gave a short LP-rounding framework for hitting axis-parallel segments with weighted points. The algorithm improves the immediate factor-2 bound by rounding one orientation exactly and repairing the residual subproblem on the other orientation using exact one-dimensional optimization. This yields a deterministic $(1+2/e)$-approximation for the weighted problem and a deterministic $(1 + 1/(e-1))$-approximation in the unweighted case. For bounded-complexity subclasses, local search yields a PTAS for the Hitting Set problem, and we also extend our results for segments in d orientations.

Two natural directions remain open. First, it would be interesting to determine whether the analysis can be strengthened further, or whether these bounds are close to the true integrality behavior of the natural LP. For the Set Cover and Independent Set problems, the best known approximation is the straightforward factor-2 bound. The integrality gap of the natural LP-formulation for Independent Set was shown to be 2 [9]. Hence, breaking the factor-2 for the independent set problem cannot go via rounding the natural LP and new ideas are required. Finally, the approach suggests exploring whether similar one-orientation-first rounding schemes can be extended to richer families such as thin rectangles.

References

1 Anna Adamaszek, Sariel Har-Peled, and Andreas Wiese. Approximation schemes for independent set and sparse subsets of polygons. *Journal of the ACM*, 66(4):29:1–29:40, 2019. doi:10.1145/3326122.

2 Noga Alon. A non-linear lower bound for planar epsilon-nets. *Discret. Comput. Geom.*, 47(2):235–244, 2012. doi:10.1007/S00454-010-9323-7.

3 Boris Aronov, Esther Ezra, and Micha Sharir. Small-size ϵ-nets for axis-parallel rectangles and boxes. *SIAM J. Comput.*, 39(7):3248–3282, 2010. doi:10.1137/090762968.

4 Rom Aschner, Matthew J. Katz, Gila Morgenstern, and Yelena Yuditsky. Approximation schemes for covering and packing. In Subir Kumar Ghosh and Takeshi Tokuyama, editors, *WALCOM: Algorithms and Computation, 7th International Workshop, WALCOM 2013, Kharagpur, India, February 14-16, 2013. Proceedings*, volume 7748 of *Lecture Notes in Computer Science*, pages 89–100. Springer, 2013. doi:10.1007/978-3-642-36065-7_10.

5 Jozsef Balogh and Jozsef Solymosi. On the number of points in general position in the plane. *Discrete Analysis*, pages 1–20, 2018. doi:10.19086/da.4438.

6 Aritra Banik, Rajiv Raman, and Saurabh Ray. Geometric optimization parameterized by piercing complexity. In *EATCS International Colloquium on Automata, Languages, and Programming (ICALP) 2026*, 2026. to appear.

7 Aniket Basu Roy, Sathish Govindarajan, Rajiv Raman, and Saurabh Ray. Packing and covering with non-piercing regions. *Discrete & Computational Geometry*, 2018.

8 Hervé Brönnimann and Michael T. Goodrich. Almost optimal set covers in finite VC-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995. doi:10.1007/BF02570718.

9 Marco Caoduro, Jana Cslovjecsek, Michal Pilipczuk, and Karol Węgrzycki. On the independence number of intersection graphs of axis-parallel segments. *J. Comput. Geom.*, 14(1):144–156, 2023. doi:10.20382/JOCG.V14I1A5.

10 Timothy M. Chan and Elyot Grant. Exact algorithms and APX-hardness results for geometric packing and covering problems. *Computational Geometry*, 47(2):112–124, 2014. doi:10.1016/j.comgeo.2012.04.001.

11 Timothy M. Chan and Elyot Grant. Exact algorithms and APX-hardness results for geometric packing and covering problems. *Computational Geometry Theory and Applications*, 47(2):112–124, 2014. doi:10.1016/J.COMGEO.2012.04.001.

12 Timothy M. Chan, Elyot Grant, Jochen Könemann, and Malcolm Sharpe. Weighted capacitated, priority, and geometric set cover via improved quasi-uniform sampling. In *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*, Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA, pages 1576–1585, 2012. doi:10.1137/1.9781611973099.125.

13 Timothy M. Chan and Sariel Har-Peled. Approximation algorithms for maximum independent set of pseudo-disks. *Discrete and Computational Geometry*, 48(2):373–392, 2012. doi:10.1007/S00454-012-9417-5.

14 Jana Cslovjecsek, Michal Pilipczuk, and Karol Węgrzycki. Parameterized approximation for maximum weight independent set of rectangles and segments. In Timothy M. Chan, Johannes Fischer, John Iacono, and Grzegorz Herman, editors, *32nd Annual European Symposium on Algorithms, ESA 2024, Royal Holloway, London, United Kingdom, September 2-4, 2024*, LIPIcs, pages 43:1–43:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ESA.2024.43.

15 Rong-chii Duh and Martin Fürer. Approximation of k-set cover by semi-local optimization. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 256–264. ACM, 1997. doi:10.1145/258533.258599.

16 Guy Even, Dror Rawitz, and Shimon Shahar. Hitting sets when the vc-dimension is small. *Inf. Process. Lett.*, 95(2):358–362, 2005. doi:10.1016/j.ipl.2005.03.010.

17 Sándor P. Fekete, Kan Huang, Joseph S. Mitchell, Ojas Parekh, and Cynthia A. Phillips. Geometric hitting set for segments of few orientations. *Theor. Comp. Sys.*, 62(2):268–303, February 2018. doi:10.1007/s00224-016-9744-7.

18 Jacob Fox and János Pach. Applications of a new separator theorem for string graphs. *Comb. Probab. Comput.*, 23(1):66–74, 2014. doi:10.1017/S0963548313000412.

19 Sariel Har-Peled and Kent Quanrud. Approximation algorithms for polynomial-expansion and low-density graphs. In *Algorithms – ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14–16, 2015, Proceedings*, pages 717–728, 2015. doi:10.1007/978-3-662-48350-3_60.

20 David Haussler and Emo Welzl. epsilon-nets and simplex range queries. *Discrete & Computational Geometry*, 2:127–151, 1987. doi:10.1007/BF02187876.

21 J Håstad. Clique is hard to approximate within $n^{1-\varepsilon}$. *Acta Mathematica*, 182(1):105–142, 1999.

22 Katarzyna Kowalska and Michał Pilipczuk. Parameterized and approximation algorithms for coverings points with segments in the plane. In *41st International Symposium on Theoretical Aspects of Computer Science*, 2024.

23 Richard J. Lipton and Robert E. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.

24 Dániel Marx. Parameterized complexity of independence and domination on geometric graphs. In *2nd International Workshop on Parameterized and Exact Computation (IWPEC 2006)*, volume 4169 of *Lecture Notes in Computer Science*, pages 154–165. Springer, 2006. doi:10.1007/11847250_14.

25 Jirí Matousek, Raimund Seidel, and Emo Welzl. How to net a lot with little: Small epsilon-nets for disks and halfspaces. In Raimund Seidel, editor, *Proceedings of the Sixth Annual Symposium on Computational Geometry, Berkeley, CA, USA, June 6-8, 1990*, pages 16–22. ACM, 1990. doi:10.1145/98524.98530.

26 Nabil H. Mustafa. *Sampling in combinatorial and geometric set systems*, volume 265. American Mathematical Society, 2022.

27 Nabil H. Mustafa, Rajiv Raman, and Saurabh Ray. Quasi-polynomial time approximation scheme for weighted geometric set cover on pseudodisks and halfspaces. *SIAM Journal on Computing*, 44(6):1650–1669, 2015. doi:10.1137/14099317X.

28 Nabil H. Mustafa and Saurabh Ray. Improved results on geometric hitting set problems. *Discrete & Computational Geometry*, 44(4):883–895, 2010. doi:10.1007/S00454-010-9285-9.

29 János Pach and Gábor Tardos. Tight lower bounds for the size of epsilon-nets. In *Proceedings of the twenty-seventh annual symposium on Computational geometry*, pages 458–463, 2011. doi:10.1145/1998196.1998271.

30 János Pach and Géza Tóth. Graphs drawn with few crossings per edge. *Comb.*, 17(3):427–439, 1997. doi:10.1007/BF01215922.

31 Evangelia Pyrga and Saurabh Ray. New existence proofs for ϵ-nets. In *Proceedings of the Twenty-fourth Annual Symposium on Computational Geometry*, SoCG '08, pages 199–207, New York, NY, USA, 2008. ACM.

32 Rajiv Raman and Saurabh Ray. Constructing planar support for non-piercing regions. *Discrete and Computational Geometry*, 64(3):1098–1122, 2020. doi:10.1007/S00454-020-00216-W.

33 Rajiv Raman, Siddhartha Sarkar, and Jatin Yadav. Hitting axis-parallel segments with weighted points, 2026. doi:10.48550/arXiv.2605.14499.

34 Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.

35 Moshe Shaked and J. George Shanthikumar. *Stochastic Orders*. Springer Series in Statistics. Springer New York, New York, NY, 2007. doi:10.1007/978-0-387-34675-5.

36 Kasturi Varadarajan. Weighted geometric set cover via quasi-uniform sampling. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*, pages 641–648, 2010. doi:10.1145/1806689.1806777.

Length-Constrained Network Design in Planar Digraphs

Chandra Chekuri
Siebel School of Computing and Data Science, University of Illinois, Urbana-Champaign, IL, USA

Rhea Jain
Siebel School of Computing and Data Science, University of Illinois, Urbana-Champaign, IL, USA

Abstract

We study *length-constrained* generalizations of Directed Steiner Tree (DST) and Directed Steiner Forest (DSF) in planar digraphs. In both problems, the input is a directed graph with edge costs. DST asks for a min-cost subgraph connecting a root to a given set of terminals, and DSF asks for a min-cost subgraph connecting each of a given set of source-sink terminal pairs. In the length-constrained setting, each edge has both a cost and a length, and the input includes a length bound h; the goal is to find a min-cost subgraph connecting each terminal pair via a path of length at most h. Our work is motivated by a recent line of results showing that several network design problems that are traditionally hard in directed graphs admit polylogarithmic approximation ratios in planar digraphs. We give polylogarithmic bicriteria approximation algorithms for length-constrained analogues of DST and DSF in planar digraphs. Our approximation ratios match the best known for DST and DSF in planar digraphs, with an $O(\log k)$ violation of the length constraint, where k denotes the number of terminals (or terminal pairs). As corollaries, we obtain polylogarithmic approximations for *buy-at-bulk* DST and DSF in planar digraphs.

2012 ACM Subject Classification Theory of computation → Routing and network design problems; Theory of computation → Discrete optimization

Keywords and phrases Approximation algorithms, network design, directed graphs, length constraints

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.15

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2607.25811

Funding Chandra Chekuri and Rhea Jain are partially supported by NSF grant CCF-2402667.

Acknowledgements We thank David Zheng for helpful discussion on length-constrained Directed Steiner Forest.

1 Introduction

Directed Steiner Tree (DST) and *Directed Steiner Forest* (DSF) are fundamental problems in network design with a wide array of applications. For both problems, the input consists of a directed graph $G = (V, E)$ with edge costs $c : E \to \mathbb{R}_+$. In Directed Steiner Tree, given a root $s \in V$ and terminals $S \subseteq V \setminus \{s\}$, the goal is to find a min-cost subgraph containing an s-t path for each $t \in S$. Directed Steiner Forest generalizes DST to the *multicommodity* setting: given a set of demand pairs $D = \{(s_i, t_i)\}_{i \in [k]}$, the goal is to find a min-cost subgraph containing an s_i-t_i path for each $i \in [k]$.

DST and DSF are NP-hard, and in fact have strong lower bounds on their approximability. DST generalizes several hard problems in combinatorial optimization, such as Set Cover and Group Steiner Tree. It is hard to approximate to an $\Omega(\log^2 k / \log \log k)$-factor under plausible complexity assumptions [33], and to a slightly weaker $\Omega(\log^{2-\epsilon} k)$-factor unless


© Chandra Chekuri and Rhea Jain;
licensed under Creative Commons License CC-BY 4.0

NP is contained in randomized quasi-poly time [37]; here, k denotes $|S|$. The best known approximation ratios are $O(\log^2 k/\log\log k)$ in *quasi-polynomial* time [33, 31, 10] and $O(k^\epsilon)$ for any fixed $\epsilon > 0$ in polynomial time [60]. Whether DST admits a polylogarithmic approximation in polynomial time is a major open question. We remark that the natural LP relaxation for DST has polynomial-factor integrality gap [61, 51]; this is one reason that developing improved approximations for DST has proven challenging. DSF is hard to approximate to a factor $\Omega(2^{\log^{1-\epsilon}(n)})$ for any $\epsilon > 0$ and thus does not admit a polylogarithmic approximation [24]. The best known approximation ratios are $O(k^{\frac{1}{2}+\epsilon})$ [12] and $O(n^{2/3+\epsilon})$ [7]. An exciting recent line of work has shown that many problems that are hard to approximate in general directed graphs are tractable when the input graph is *planar* [45, 29, 17, 14]. In planar digraphs, DST admits an $O(\log k)$-approximation [29], a natural LP relaxation has integrality gap $O(\log^2 k)$ [17], and DSF admits an $O(\log^6 k)$-approximation [14].

In this paper, we consider a generalization of DST and DSF to the *length-constrained* setting. As the name suggests, these impose an additional constraint on the length of paths used to connect terminal pairs. Formally, in *Length-Constrained Directed Steiner Tree* (LC-DST) and *Length-Constrained Directed Steiner Forest* (LC-DSF), we are given an additional *length* function on the edges $\ell : E \to \mathbb{R}_+$ and a length constraint h; the goal is to find a min-cost subgraph in which terminal pairs are connected via a path of length at most h. There are several practical reasons why connectivity alone may not suffice: short paths are desirable for faster transportation and communication, improved tolerance to failure, and more. Furthermore, length-constrained network design is closely related to several other important problems in combinatorial optimization. For instance, *buy-at-bulk* network design (closely related to cost-distance and other problems) is a well-studied and practical model which places a penalty on the path lengths in the objective function instead of imposing strict bounds on path lengths. Progress in length-constrained network design has led to the resolution of several open problems in buy-at-bulk network design [13, 16]. As a result of its practical and theoretical importance, length-constrained network design has been very well studied; see Section 1.3 for details.

In undirected graphs, adding length constraints to problems adds significant complexity to a problem. As an example, consider the minimum spanning tree problem; this is easily solvable in polynomial time. However, the length-constrained minimum spanning tree problem is essentially equivalent in approximability to DST (see [39] for a formal reduction). A common way to handle this difficulty is by relaxing the length constraint, allowing for some slack in both the length and the cost (see Definition 1). In directed graphs, on the other hand, length constraints do not add inherent complexity. A simple reduction shows that LC-DST is equivalent to DST, as shown in Figure 1,[1] and one can define a similar reduction from LC-DSF to DSF. However, these reductions do not preserve planarity. Given the fruitful progress on network design problems in planar digraphs, it is natural to ask if LC-DST and LC-DSF admit polylogarithmic approximations in planar digraphs. We answer these questions affirmatively.

1.1 Our Results

We obtain polylogarithmic *bicriteria* approximation algorithms for LC-DST and LC-DSF in planar digraphs.

[1] For polynomial edge lengths one can replace each edge e with a path of length $\ell(e)$. For larger edge lengths, one can obtain a similar reduction losing a $(1+\epsilon)$ factor in length slack.

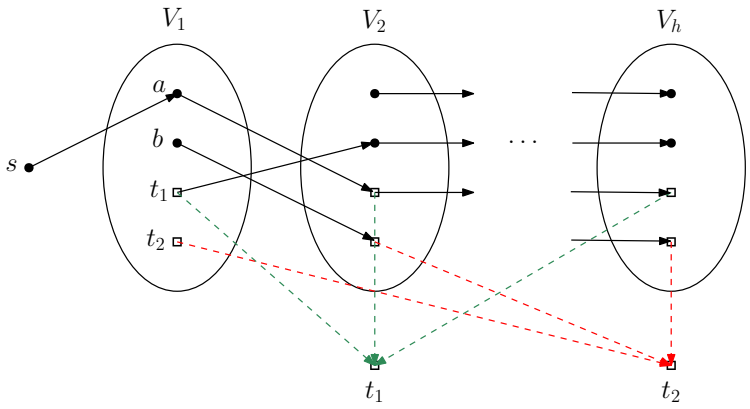


Figure 1 This example assumes all edge lengths are 1. Suppose the input graph is a path s, a, t_1, b, t_2 with terminals t_1, t_2. We create h copies of the vertex set $V_1, \ldots, V_h$, and for every $(u, v) \in E$ and all $i < h$, we add an edge from the copy of u in V_i to that of v in V_{i+1}. The terminals in the resulting DST instance are given at the bottom of the figure.

▶ **Definition 1.** *An (α, β)-bicriteria approximation for a length constrained network design problem with length constraint h is a solution $F \subseteq E$ in which all demand pairs are connected via a path of length at most $\beta \cdot h$, and $c(F) \leq \alpha \cdot \mathrm{OPT}$, where* OPT *is the cost of an optimal solution. We refer to α as the* approximation ratio *and β as the* length slack*.*

The approximation factors for all of our results match the best known ratios for DST and DSF in planar digraphs without length constraints; all algorithms incur an $O(\log k)$ factor in the length slack. We remark that this $O(\log k)$ length slack factor is essentially the best known for many problems in length-constrained network design when one seeks a polylogarithmic approximation ratio, including length-constrained MST in undirected graphs. There are known algorithms for length-constrained MST with polynomial approximation ratio that do not lose any length slack. Reducing the length slack to $O(1)$ while ensuring polylogarithmic approximation ratio is an enticing research direction with some partial progress discussed further in Section 1.3. We highlight one such recent work: in undirected planar graphs, [39] obtains an $O(\log^{1+\epsilon}(n))$-approximation with $(1+\epsilon)$ length slack. However, it is unknown if this approach would extend to the directed setting, and there appear to be several challenges.

We first consider length-constrained Directed Steiner Tree. An approximation algorithm is *LP-competitive* if the cost of the given solution is at most α times the value of an optimum *fractional* solution to a natural LP relaxation. We describe an LP relaxation to LC-DST in Section 3. We obtain two algorithms: the first (Theorem 2) is with respect to the optimum integral solution, and the second (Theorem 3) is LP-competitive.

▶ **Theorem 2.** *There is a bicriteria $(O(\log k), O(\log k))$-approximation for LC-DST in planar digraphs.*

▶ **Theorem 3.** *There is a bicriteria $(O(\log^2 k), O(\log k))$ LP-competitive approximation for LC-DST in planar digraphs.*

Next, we consider length-constrained Directed Steiner Forest. Note that Theorem 4 provides an approximation algorithm with respect to the optimum integral solution; obtaining an LP-competitive approximation remains open. We note that an LP-competitive approximation for planar DSF is not known even without length constraints.

▶ **Theorem 4.** *There is a bicriteria $(O(\log^6 k), O(\log k))$-approximation for Length-Constrained Directed Steiner Forest in planar digraphs.*

1.1.1 Extensions

We mention two further extensions of our work. Details and formal theorem statements are omitted as they are technically involved and require new machinery, see [41] for details.

Buy-at-bulk network design. As mentioned earlier, there are known connections between buy-at-bulk and length-constrained network design problems. Prior work in *undirected* graphs [13, 16] demonstrated an LP-based reduction from a buy-at-bulk problem to its corresponding length-constrained problem. The resulting approximation ratio for the buy-at-bulk problem depends on both the approximation and length slack of the algorithm for the length-constrained problem. These arguments extend to directed graphs; thus, one can obtain a polylogarithmic approximation for buy-at-bulk DST in planar digraphs as a corollary of Theorem 3. In the multicommodity setting, more effort is needed, since Theorem 4 does not yield an LP-competitive solution. Nevertheless, one can obtain a polylogarithmic approximation for buy-at-bulk DSF in planar digraphs by arguing that there exists a low-density *junction tree*. Similar arguments have been used in [11, 4, 13].

Group, Covering, and Polymatroid Steiner Tree. Chekuri et al. [17] developed polylogarithmic approximation algorithms for several rooted generalizations of Directed Steiner Tree in planar digraphs. This includes the *Directed Polymatroid Steiner Tree* problem, where a polymatroid is defined on the vertex set, and a feasible tree is required to span a base of the polymatroid. A polymatroid is an integer-valued, normalized, monotone, submodular function. This generalizes several important network design problems, including Group and Covering Steiner Tree. At a high level, the algorithm of [17] explicitly constructs the recursion tree of the divide-and-conquer algorithm of [29]. One can view this recursion tree as a "tree embedding" suitable for solving rooted problems, allowing one to reduce to solving the problem on a tree. One can similarly construct a "tree embedding" that respects length constraints, based on the divide-and-conquer algorithm of Theorem 2. Thus, one can obtain polylogarithmic approximations for length-constrained rooted variants of directed polymatroid, covering, and group problems.

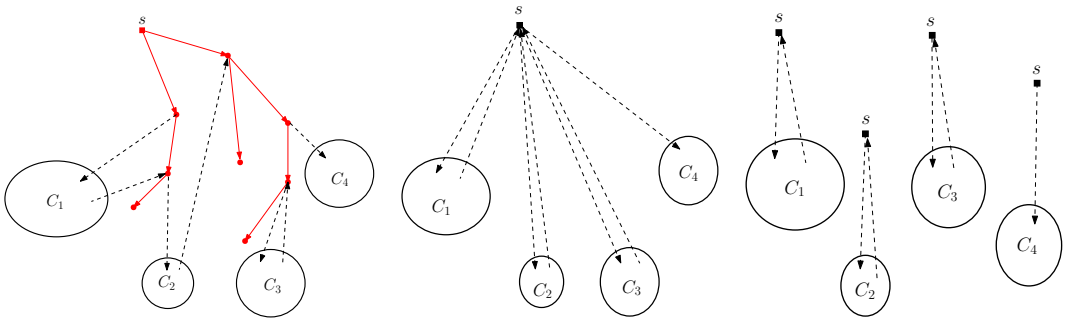
1.2 Technical Overview

The proofs of Theorem 2 and Theorem 3 on LC-DST adapt established ideas to the length-constrained setting. The main novel contribution of this work is Theorem 4 on LC-DSF, as the previous approach for DSF in planar digraphs fails to extend. We outline the ideas for all three algorithms here.

Approximation for DST in planar digraphs. We start by giving an overview of the divide-and-conquer $O(\log k)$-approximation given by Friggstad and Mousavi [29]. The algorithm uses Thorup's shortest path separator theorem applied to directed graphs:

▶ **Lemma 5** ([29, 58]). *Let G be a planar directed graph with non-negative edge costs $c(e)$, non-negative vertex weights $w(v)$, and a root $s \in V$ such that every vertex in V is reachable from s. There exists a polynomial time algorithm to find three shortest dipaths P_1, P_2, P_3 starting at s such that every weakly connected component of $G \setminus (P_1 \cup P_2 \cup P_3)$ has at most half the vertex weight of G.*

The high-level idea in [29] is simple; see Figure 2. Suppose we can guess the optimum solution value OPT for a given DST instance. Then, we can remove all vertices farther than OPT from s and use Lemma 5 to find three paths, each of cost at most OPT, whose



(a) Step 1. **(b)** Step 2. **(c)** Step 3.

Figure 2 High-level overview of the divide-and-conquer algorithm. In Step 1, we compute the planar separator (shown in red). Step 2 demonstrates contracting the separator into the root, and Step 3 demonstrates the resulting independent sub-instances.

removal results in connected components each containing at most half the terminals. We shrink the paths into s and recurse on the resulting sub-instances. The recursion depth is $O(\log k)$, which bounds the total cost to $O(\log k) \cdot \mathrm{OPT}$. Implementing the guess of OPT in each recursive call is challenging; naively guessing requires quasipolynomial time. [29] obtain a polynomial time algorithm by a refined argument where the guess for OPT is folded into the recursion itself.

Handling Length-Constraints and Bounding Integrality Gap. The main challenge in using the approach of [29] in *length-constrained* Directed Steiner Tree lies in defining the shortest path tree on which to apply the planar separator result. Ideally, one would like for the tree to capture length-constrained distances, where the h-length-constrained distance from u to v is the min-cost over all u-v paths of length at most h. Unfortunately, length-constrained distances do not form a metric, so one cannot use a standard shortest path tree. To circumvent this issue, we use the well-known "mixture-metric", which balances length constraints and costs. This suffices to obtain Theorem 2. We remark that recent work of [39] on length-constrained MST in *undirected planar graphs* also uses the mixture-metric to apply the separator lemma. To prove Theorem 3, we use an idea introduced in [17], which uses the LP optimum value as the estimate for OPT instead of guessing. The algorithm of [17] extends to the length-constrained setting with the mixture-metric in a similar fashion.

Directed Steiner Forest via Junction Trees. To tackle the multicommodity setting, we build on the *junction tree* based algorithm of [14] for planar DSF. At a high level, junction trees allow one to reduce a multicommodity problem to its single-source counterpart. This technique has been used in a variety of network design problems; it was first explicitly described in the context of non-uniform buy-at-bulk network design [36, 11], though it was implicitly used earlier for Directed Steiner Forest in [10]. The idea is to repeatedly find well-structured *low-density partial solutions*, where a partial solution is feasible for a subset of terminal pairs, and the density is the ratio of cost to the number of terminal pairs connected. Using a standard iterative approach for covering problems, one can reduce a multicommodity problem to its corresponding min-density partial solution problem, losing an $O(\log k)$ factor in the approximation ratio. We aim to find partial solutions that contain a "junction" vertex $v \in V$ through which many pairs connect. This allows us to view the problem as a single-source problem with source v. The approximation for DSF in planar

digraphs of [14] consists of two steps: (1) proving the existence of a low-density junction tree, and (2) efficiently finding an approximate min-density junction tree. The second step uses a bucketing-and-scaling technique introduced by [11], which relies on the existence of an LP-competitive single-source algorithm. With Theorem 3, this extends with little difficulty to the length-constrained setting. Our main technical contribution is the first step: showing existence of a low-density junction tree, as length constraints pose several new challenges, and the proof of [14] does not cleanly extend. We defer further discussion to Section 4.

1.3 Related Work

Directed Steiner Tree. Zelikovsky [60] was the first to address the approximability of DST. He obtained an $O(k^\epsilon)$-approximation for any fixed $\epsilon > 0$ via two ideas. He developed a recursive greedy algorithm and analyzed its performance as a function of recursion depth. He then showed that one can reduce the problem on a general directed graph to a problem on a depth d DAG (via the transitive closure of the original graph) at the loss of an approximation factor that depends on d. Charikar et al. [10] refined the algorithm and analysis in [60] and combined it with the depth reduction. They showed that one can obtain an $O(d^2 k^{1/d} \log k)$ approximation in $O(n^{O(d)})$-time; this led to an $O(\log^3 k)$-approximation in quasi-polynomial time. Subsequently, Grandoni et al. [33] improved the approximation to $O(\log^2 k / \log\log k)$ in quasi-polynomial time via a more sophisticated LP-based approach. A different approach that also yields the same bound was given by Ghuge and Nagarajan [31], and this is based on a refinement of the recursive greedy algorithm for walks in graphs [18]. The approach of [31] extends to several related problems, including directed single-source buy-at-bulk.

Zosin and Khuller [61] showed that the natural cut-based LP relaxation for DST has an integrality gap of $\Omega(\sqrt{k})$. However, their example only showed a gap of $\Omega(\log n)$ as a function of the number of nodes n. There was some hope that the integrality gap is polylogarithmic in n; however, [51] recently showed that the gap is $\Omega(n^\delta)$ for some $\delta > 0$ by modifying the construction in [61]. Interestingly, these lower bound examples are DAGs with $O(1)$-layers, for which the recursive-greedy algorithm yields an $O(\log k)$-approximation in polynomial-time. Rothvoss [57] showed that $O(\ell)$-levels of the Lasserre SDP hierarchy when applied to the standard cut-based LP reduces the integrality gap to $O(\ell \log k)$ on DAGs with ℓ layers. This was later refined to show that $O(\ell)$-levels of the Sherali-Adams hierarchy suffices [27]. However, both these approaches also require quasi-polynomial time to obtain a polylogarithmic approximation. Recent work by Laekhanukit [50] shows a flow-based LP relaxation that has polylogarithmic integrality gap under some strong assumptions on the structure of an optimal fractional solution.

Directed Steiner Forest. The first nontrivial approximation for Directed Steiner Forest was an $\tilde{O}(k^{2/3})$-approximation given by Charikar et al. [10]. This follows a similar iterative density-based procedure as the *junction tree* approach; however, they restrict to trees of a much simpler structure. This approximation ratio was subsequently improved to $O(k^{\frac{1}{2}+\epsilon})$ by Chekuri et al. [12]; they showed that given an instance (G, D) of DSF, there exists a junction tree of density at most $O(k^{1/2})$ times the optimum. They then provided an algorithm to find a low-density junction tree via height reduction and Group Steiner Tree rounding. DSF has improved approximation ratios when k is large. Feldman, Kortsarz, and Nutov [25] obtained an $O(n^\epsilon \cdot \min(n^{4/5}, m^{2/3}))$-approximation using a junction-based approach. This analysis was refined by Berman et al. [7] using ideas developed for finding directed spanners, giving an improved approximation ratio of $O(n^{2/3+\epsilon})$. For DSF with *uniform* edge costs, Chlamtáč et al. obtained an $O(n^{3/5+\epsilon})$-approximation [19], which was subsequently improved to $O(n^{4/7+\epsilon})$ by Abboud and Bodwin [1].

Length-Constrained Network Design. Length-constrained network design has been primarily studied in undirected graphs. It was first introduced by Balakrishnan and Altinkemer [5], and has since been studied in both approximation and fast exact algorithms. The length-constrained s-t shortest path problem is solvable exactly in polynomial time when edge lengths are polynomially bounded, via the textbook Bellman-Ford dynamic programming algorithm. However, it is NP-hard for arbitrary lengths [42].[2] Moreover, length-constrained MST is NP-hard even in the special case of hop constraints.[3] From an approximation algorithms standpoint, until recently, the majority of the work on length-constrained network design had been limited to single-source settings, such as s-t shortest path [43, 59, 38, 32, 53, 8], MST [21, 2, 55, 3, 47, 56, 54, 40], Steiner Tree [49, 44], and k-Steiner Tree [36, 46]. While s-t shortest path admits an FPTAS [38], other length-constrained problems have strong hardness results under *strict* length constraints. Length-constrained MST is as hard as Directed Steiner Tree, and the length-constrained variant of Steiner forest has no $o(2^{\log^{1-\epsilon}(n)})$-approximation [23]. These problems are more tractable when length constraints are relaxed. A simple merging algorithm [54] yields an $(O(\log n), O(\log n))$-bicriteria approximation for length-constrained MST and Steiner Tree. Recent progress by Hershkowitz and Huang allows one to reduce the length slack to $O(\log n/\log\log n)$ while increasing the approximation ratio by a polylogarithmic factor [40]. The multicommodity setting has been more challenging. For length-constrained Steiner Forest, polylogarithmic bicriteria approximations were implicitly known via buy-at-bulk [11, 48]. There has also been a line of work on hop-constrained *tree embeddings*, which have led to bicriteria approximations for hop-constrained Steiner Forest and various related problems [35, 26]. Chekuri and Jain [13], building on hop-constrained oblivious routing tools of Ghaffari, Haeupler, and Zuzic [30], obtained LP-competitive algorithms for multicommodity problems in buy-at-bulk and length-constrained settings. In directed graphs, a recent work obtains polynomial factor approximations for Directed Buy-at-Bulk and related spanner problems [34]; these approximation ratios essentially match the best known factors for DSF in general digraphs.

Planar and Minor-Free Graphs. Improved approximation ratios have been obtained for several problems in special classes of graphs, such as planar and minor-free graphs. We first discuss undirected graphs. In planar graphs, Steiner Tree admits a PTAS [9]; this was later extended to a PTAS for Steiner Forest in graphs of bounded genus [6]. Recently, [20] obtained a QPTAS for Steiner Tree in minor-free graphs. Furthermore, although the node-weighted variant of Steiner Tree captures Set Cover in general graphs, there exists a constant factor approximation in planar graphs, and more generally, in any proper minor-closed graph family [22]. In directed graphs, several recent results have built on tools given by Thorup [58] to obtain approximation algorithms in planar digraphs. Kawarabayashi and Sidiropoulos [45] gave a polylogarithmic upper bound on the multicommodity flow-cut gap in planar digraphs; this was derandomized and extended to the node-weighted setting by [15]. As discussed earlier, Friggstad and Mousavi [29] obtained an $O(\log k)$-approximation for DST in planar digraphs. This was extended by Chekuri et al. [17] to an LP-competitive $O(\log^2 k)$-approximation. Chekuri et al. [17] also obtained polylogarithmic approximations for several rooted variants of DST in planar digraphs, such as group, covering, polymatroid, and multirooted versions. Chekuri and Jain [14] then obtained an $O(\log^6 k)$-approximation for DSF in planar digraphs. In minor-free input graphs that are *quasi-bipartite*, Friggstad and Mousavi [28] obtained a constant-factor approximation for DST.[4]

[2] The length-constrained shortest path problem is often referred to as *Restricted Shortest Path*.
[3] Hop constraints refer to the unit-edge-length setting of length-constrained network design.
[4] The input is quasi-bipartite if there are no edges between any two non-terminal nodes.

2 Preliminaries

2.1 Notation

Let $G = (V, E)$ be a directed graph with edge costs $c : E \to \mathbb{R}_+$. For $E' \subseteq E$, we denote $c(E') = \sum_{e \in E'} c(e)$. We assume all edge costs $c(e) \geq 1$ and are polynomially bounded in n. This can be done with a $(1 + o(1))$ factor loss in approximation, by guessing the cost of the optimal solution OPT, contracting edges with cost much smaller than OPT, and scaling appropriately.

For $S \subseteq V$, we use $E[S]$ to denote the set of edges of E with both endpoints in S, $G[S]$ to denote the subgraph $(S, E[S])$ *induced by* S in G, and $\delta^+(S) = \{(u, v) \in E : u \in S, v \notin S\}$ to denote the *out-cut* of S. The *height* of an out-tree T rooted at s is the maximum over all $v \in V(T)$ of the number of edges on the unique s-v path. The *size* of the tree is the number of vertices in the tree. To *contract* an edge (u, v) is to merge its endpoints into a single vertex w_{uv}, replacing all edges incident to u or v with edges incident to w_{uv} and removing resulting self-loops. For a subgraph $G' \subseteq G$, we use G/G' to denote the graph obtained by contracting every edge of G' and $G - G'$ to denote the graph obtained by deleting every edge in G'. A graph H obtained from G via edge contractions and edge or vertex deletions is called a *minor* of G. It is easy to see that if G is planar, then any minor of G is planar as well. A weakly connected component of G is a maximal connected component of the underlying undirected graph obtained from G by ignoring the edge orientations. We call this undirected graph the *undirected version* of G.

The length $\ell(p)$ of a path p is the sum of the lengths of its edges. More generally, we write $\ell(F) := \sum_{e \in F} \ell(e)$ for any $F \subseteq E$. For any subset of edges $F \subseteq E$ and any pair of vertices $u, v \in V$, we let $\ell_F(u, v)$ denote the length of the shortest-length u-v path in F. We generalize this notation to sets: for $A, B \subseteq V$, $\ell_F(A, B)$ is the length of the shortest-length path in F from A to B. We drop F when it is clear from context. For a tree T and vertices $u, v \in V(T)$, we write $P_T(u, v)$ to denote the unique u-v path in T.

2.2 Mixture Metric

We write $d^{(h)}(s, t)$ to denote the cost of the shortest h-length constrained s-t path. Computing $d^{(h)}(s, t)$ is NP-hard [42]; however, it can be approximated to a $(1 + \epsilon)$-factor [38]. Unlike traditional graph distances, $d^{(h)}$ does not form a metric. We instead consider the *mixture metric*; this is a standard notion to facilitate working with length constraints, and was highlighted in recent work in length-constrained network design by [35].

Given a length constraint h and a cost bound γ, we define the mixture metric as follows: for each $e \in E$, the *mixture cost* function is $c'(e) = c(e) + \frac{\gamma}{h}\ell(e)$. For all $s, t \in V$, let $d'(s, t)$ be the cost of the shortest s-t path with respect to c'. It is not difficult to see that d' forms a metric. Claim 6 demonstrates how d' approximates $d^{(h)}$; the proof is simple and deferred to a full version.

▷ Claim 6. Let d' be the mixture metric with length constraint h and cost bound γ.

1. $d'(s, t) \leq d^{(h)}(s, t) + \gamma$;
2. $d^{(h')}(s, t) \leq d'(s, t)$, where $h' = \frac{d'(s,t)}{\gamma} \cdot h$.

2.3 Planar Separator

The key property of planarity used in our algorithms is the fact that planar graphs have good shortest path separators, as shown in Lemma 5. We state a more general lemma on *undirected graphs*. This lemma, essentially proved by Lipton and Tarjan [52], but given

explicitly by Thorup [58], states that every spanning tree on any planar graph contains three paths whose removal results in connected components each containing at most half the number of vertices. The formal statement is as follows.

▶ **Lemma 7** ([52, 58]). *Let G be an undirected connected graph with non-negative edge costs $c(e)$, non-negative vertex weights $w(u)$, and a spanning tree T rooted at $v \in V$. There exists a polynomial time algorithm to find three vertices $u_1, u_2, u_3 \in V$ such that each component of $G \setminus (P_T(v,u_1) \cup P_T(v,u_2) \cup P_T(v,u_3))$ has at most half the vertex weight of G.*

Note that one can recover the directed variant (Lemma 5) by considering the spanning tree T to be a shortest path tree rooted at v.

3 Length-Constrained Directed Steiner Tree

In this section, we consider length-constrained Directed Steiner Tree. We provide an $O(\log k)$-approximation in Section 3.1, and an LP-based $O(\log^2 k)$-approximation in Section 3.2.

First, we define a subroutine PruneAndSeparate (Algorithm 1) which will be used in both the integral and LP-based algorithms. This subroutine takes as input a graph $G = (V, E)$ with edge costs $c(e)$, a root $s \in V$, terminals $S \subseteq V$, and a guess γ for the cost of the optimal solution. PruneAndSeparate(G, s, S, γ) removes all vertices further than γ away from s, and uses Lemma 5 on the resulting graph with vertex weights set to 1 on the terminals and 0 elsewhere. This yields a *planar separator* $P := P_1 \cup P_2 \cup P_3$ in which each resulting component of $G \setminus P$ has at most half the terminals. The subroutine contracts P into s; each component of $G \setminus P$ corresponds to a new subinstance induced by the terminals in that component along with s. PruneAndSeparate(G, s, S, γ) returns P along with the subinstances for each component.

Algorithm 1 PruneAndSeparate(G, s, S, γ).

Delete all vertices $v \in V$ with $d_G(s, v) > \gamma$
Let $P := P_1 \cup P_2 \cup P_3$ be given by Lemma 5 with weights $w(v) = \mathbb{1}_{v \in S}$ for $v \in V(G)$
Let G_P be obtained from G by contracting P into s
Let $C_1, \ldots, C_\tau$ be components of $G \setminus P$, and let $G_i \leftarrow G_P[C_i \cup \{s\}]$
return $(P, (G_1, C_1), \ldots, (G_\tau, C_\tau))$

3.1 Approximation with respect to Integral Optimum

We follow the divide-and-conquer algorithm given by Friggstad and Mousavi [29], adapted to handle length constraints. Recall the high level approach: the algorithm branches on two cases and returns the cheaper solution. In the first branch, it recurses with a halved guess for OPT; this handles the case where OPT $\leq \gamma/2$. In the second branch, it computes a planar separator P under the *mixture cost* c', and recursively solves each resulting component independently. Using the mixture cost for the separator is the main adaptation for length constraints, which ensures that the separator paths have bounded cost *and* bounded length. For clarity, we write the algorithm as though $d^{(h)}(s,t)$ can be computed exactly; replacing this with a $(1+\epsilon)$-approximation does not affect the approximation guarantees.

Algorithm 2 h-DST(I, γ): an instance $I = (G, s, S, h)$ and a "guess" γ for OPT.

if $\exists t \in S$ such that $d^{(h)}(s,t) > \gamma$ **then**
  return Infeasible (∞)
if $|S| = 1$ **then**
  return min-cost h-length-constrained s-t path in G, where t is the terminal in S
// **Recursive Step 1:**
$F^1 \leftarrow$ h-DST$(I, \gamma/2)$
// **Recursive Step 2:**
Define mixture cost $c'(e) = c(e) + \frac{\gamma}{h}\ell(e)$
$(P, (G_1, C_1), \ldots, (G_\tau, C_\tau)) \leftarrow$ PruneAndSeparate$(G = (V, E, c'), s, S, 2\gamma)$
$F^2 \leftarrow P$
for $i \in [\tau]$ **do**
  $F_i \leftarrow$ h-DST$(I_i := (G_i, s, S \cap C_i, h), \gamma)$
  $F^2 \leftarrow F^2 \cup F_i$
return $\arg\min_{F \in \{F^1, F^2\}} c(F)$

First, we argue that the algorithm runs in polynomial time. This follows directly from [29]; the proof is deferred to a full version.

▶ **Lemma 8.** *Algorithm 2 runs in polynomial time.*

Next, we prove feasibility and bound the cost of the given solution. The proof of Lemma 11 is deferred to a full version. We start with a simple property about the planar separator.

▷ Claim 9. The planar separator P computed by h-DST(I, γ) for any feasible instance satisfies the following: (i) $c(P) \le 6\gamma$, (ii) $\ell(P) \le 6h$.

Proof. Let $P = P_1 \cup P_2 \cup P_3$ be the planar separator computed by h-DST(I, γ). Recall that each P_i, for $i \in [3]$, is a shortest path from s with respect to c'. Since all vertices $u \in V$ with $d_{G,c'}(s,u) > 2\gamma$ are removed, each path P_i must have $c'(P_i) = c(P_i) + \frac{\gamma}{h}\ell(P_i) \le 2\gamma$. Since both terms are non-negative, $c(P_i) \le 2\gamma$, so $c(P) \le 6\gamma$. Similarly, $\frac{\gamma}{h}\ell(P_i) \le 2\gamma$, so $\ell(P_i) \le 2h$ and thus $\ell(P) \le 6h$. ◁

▶ **Lemma 10.** *Let F be the solution returned by* h-DST(I, γ) *for a feasible instance (I, γ). Then, F contains an s-t path of length at most $O(\log |S|) \cdot h$ for each terminal $t \in S$.*

Proof. We prove by induction on $\gamma + |S|$ that if the instance (I, γ) is feasible, then $F :=$ h-DST(I, γ) contains a path from s to t of length at most $6h(\log |S| + 1)$, for each $t \in S$. The base case is clear; if there is one terminal, then F is an s-t path of length h.

We consider the two recursive cases separately. For the first recursive case, if $(I, \gamma/2)$ is infeasible, then $c(F^1)$ is infinite and $F \neq F^1$. Else, by induction, F^1 contains an s-t path of length at most $6h(\log |S| + 1)$ for each $t \in S$.

Next, we consider the second recursive case. Fix $t \in S$. Observe that since (I, γ) is feasible, $d^{(h)}(s,t) \le \gamma$, so there exists an s-t path q of length at most h and cost at most γ. In particular, $c'(q) = c(q) + \frac{\gamma}{h}\ell(q) \le 2\gamma$, so t is not deleted in the pruning step. Therefore, t is either in the planar separator P or in some component C_i of $G \setminus P$. If $t \in P$, then P contains an s-t path of length at most $\ell(P) \le 6h$, by Claim 9. If $t \in C_i$, then since the recursive sub-instance contracts P into s, C_i contains a path from $\{s\} \cup P$ to t of length at most $6h(\log |S \cap C_i| + 1)$, by induction. Therefore, F^2 contains an s-t path of length at most

$$\ell(P) + 6h(\log |S \cap C_i| + 1) \le 6h + 6h(\log(|S|/2) + 1) = 6h + 6h \log |S| = 6h(\log |S| + 1).$$

This concludes the inductive argument and the proof of the lemma. ◀

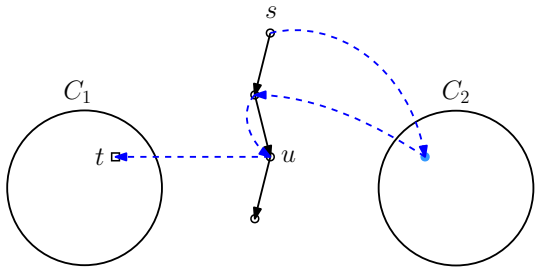


Figure 3 Consider the s-t path p shown in blue dashed lines; here the separator P is black and bolded. The portion of the path from u to t corresponds to p'.

▶ **Lemma 11.** *The cost of the solution returned by* h-DST(I, γ) *for a feasible instance* (I, γ) *is at most* $O(\log |S|) \cdot \text{OPT}$.

Theorem 2 follows from Lemma 8, Lemma 10, and Lemma 11 by applying Algorithm 2 to the given input instance I, with γ chosen to be any upper bound on OPT; e.g. $c(E)$.

3.2 LP-Based Approximation

The goal of this section is to obtain a polylogarithmic upper bound on the integrality gap of a natural LP relaxation for LC-DST in planar digraphs, defined as follows. For each terminal $t \in S$, let $\mathcal{P}_t^h$ denote the set of all s-t paths in G of length at most h. The variables x_e indicate whether an edge e is in the final solution, and variables f_p indicate how much s-t flow is sent on path p.

$$\begin{aligned} \min \quad & \sum_{e \in E} c(e) x_e \\ s.t. \quad & \sum_{p \in \mathcal{P}_t^h} f_p \geq 1 && \forall t \in S \\ & \sum_{p \in \mathcal{P}_t^h, e \in p} f_p \leq x_e && \forall e \in E, t \in S \\ & x_e, f_p \geq 0 && \forall e \in E, p \in \cup_{t \in S} \mathcal{P}_t^h \end{aligned} \tag{LC-DST-LP}$$

The LP contains an exponential number of variables but can be solved approximately (up to a $(1+\epsilon)$ factor) via separation oracle on the dual.

We provide an LP-based algorithm following a similar framework as above; here, we use the LP solution as the "guess" for OPT. This is the same approach used by [17] to obtain an LP-based approximation for DST in planar graphs. The algorithm h-frac-DST$(I, (x, f))$ is as follows, where $I = (G, s, S, h)$ is a problem instance and (x, f) is a feasible LP solution to the relaxation for LC-DST-LP.

- **Base Case.** If $S = \{t\}$, we return the min-cost h-length constrained s-t path.
- **Prune and Separate.** We set $\gamma = 2 \log |S| \sum_{e \in E} c(e) x_e$ to be the "guess" for OPT. As in Algorithm 2, we define the mixture edge cost function as $c'(e) = c(e) + \frac{\gamma}{h} \ell(e)$. Let $(P, (G_1, C_1), \ldots, (G_\tau, C_\tau))$ denote the output of PruneAndSeparate$(G = (V, E, c'), s, S, 2\gamma)$.
- **Recursive Step.** For each component $i \in [\tau]$, we define the recursive sub-instance $I_i = (G_i, s, S \cap C_i, h)$. In order to construct a feasible LP solution for I_i, we first scale up the solution; let $(\bar{x}, \bar{f}) = \left(1 + \frac{1}{\log |S|}\right)(x, f)$. We set $x_e^i = \bar{x}_e$ for all $e \in G_i$.

For the path variables, fix a terminal $t \in S \cap C_i$. For each path $p \in \mathcal{P}_t^h$ (in the graph G *before* contracting P into s), let p' denote the corresponding contracted path with resulting cycles deleted. Note that $\ell(p') \leq h$ since contraction cannot increase path length, so p' is a valid s-t path in the new sub-instance. We define $\mathcal{Q}(p')$ as the set of all paths in $\mathcal{P}_t^h$ that map to p' after contraction and cycle removal; see Figure 3. We set $f_{p'}^i = \sum_{p \in \mathcal{Q}(p')} \bar{f}_p$ for all such p' fully contained in G_i. We recursively compute $F_i = \text{h-frac-DST}(I_i, (x^i, f^i))$.

- Return $P \cup (\cup_i F_i)$

It is clear that this algorithm runs in polynomial time. We show that each recursive call is valid by showing that the constructed LP solutions are feasible for the subinstances.

▶ **Lemma 12.** *For each $i \in [\tau]$, (x^i, f^i) is a feasible LP solution on instance I_i.*

Proof. Fix $i \in [\tau]$. We show that (x^i, f^i) satisfies all LP constraints. Fix $t \in S \cap C_i$. To distinguish between the set of h-length-constrained s-t paths in the new instance I_i versus in the original input instance I, we write $\mathcal{P}_t^i$ versus $\mathcal{P}_t$; the length-constraint h is dropped from this notation for visual clarity.

We start by showing that $\sum_{p \in \mathcal{P}_t, c'(p) \leq 2\gamma} \bar{f}_p \geq 1$; that is, scaling up by a $1 + 1/\log|S|$ factor compensates for vertices deleted during the pruning step. Consider a path $p \in \mathcal{P}_t$ with $c'(p) > 2\gamma$. Since $p \in \mathcal{P}_t$, $\ell(p) \leq h$. Thus, $c(p) = c'(p) - \frac{\gamma}{h}\ell(p) > 2\gamma - \gamma = \gamma$. Observe that since (x, f) satisfies the LP constraints,

$$\sum_e c(e)x_e \geq \sum_e c(e) \sum_{p \in \mathcal{P}_t : e \in p} f_p = \sum_{p \in \mathcal{P}_t} f_p c(p) > \gamma \sum_{p \in \mathcal{P}_t : c(p) > \gamma} f_p.$$

In particular, $\sum_{p \in \mathcal{P}_t : c(p) > \gamma} f_p < \frac{1}{\gamma} \sum_e c(e)x_e = 1/(2\log|S|)$. Furthermore, since (x, f) is feasible for I, $\sum_{p \in \mathcal{P}_t} f_p \geq 1$. Therefore,

$$\sum_{p \in \mathcal{P}_t, c'(p) \leq 2\gamma} \bar{f}_p \geq (1 + \frac{1}{\log|S|}) \sum_{p \in \mathcal{P}_t : c(p) \leq \gamma} f_p \geq \left(1 + \frac{1}{\log|S|}\right)\left(1 - \frac{1}{2\log|S|}\right) \geq 1.$$

Next, we show that $\sum_{p \in \mathcal{P}_t^i} f_p^i = \sum_{p \in \mathcal{P}_t, c'(p) \leq 2\gamma} \bar{f}_p$. Let $p \in \mathcal{P}_t$, and let p' denote the corresponding contracted path. Since $c'(p) \leq 2\gamma$, no vertices of p are deleted during the pruning step. We want to show that p' is contained in G_i. Let v be the *last* vertex on p that is not in C_i; note that p must end in C_i since p is an s-t path and $t \in C_i$. Since the components $C_1, \ldots, C_\tau$ are weakly connected in $G \setminus P$, there must be no edges between them. Thus, $v \in P \cup \{s\}$. Therefore, when P is contracted into s, the path $p[s, v]$ corresponds to a walk starting and ending at the contracted supernode s. This implies that $p' = p[v, t]$, which in the contracted graph, is contained in G_i. Therefore, $p \in \mathcal{Q}(p')$ for some $p' \in \mathcal{P}_t^i$, so $\sum_{p \in \mathcal{P}_t, c'(p) \leq 2\gamma} \bar{f}_p \leq \sum_{p' \in \mathcal{P}_t^i} \sum_{p \in \mathcal{Q}(p')} \bar{f}_p = \sum_{p' \in \mathcal{P}_t^i} f_{p'}^i$, as desired. Thus, $\sum_{p \in \mathcal{P}_t^i} f_p^i \geq 1$, so the first set of LP constraints is satisfied.

Proving that the second set of LP constraints are satisfied is more straightforward. We want to show that for all edges $e \in E(G[C_i \cup \{s\}])$, $\sum_{p \in \mathcal{P}_t^i, e \in p} f_p^i \leq x_e^i$.

$$\sum_{p' \in \mathcal{P}_t^i, e \in p'} f_{p'}^i = \sum_{p' \in \mathcal{P}_t^i, e \in p'} \sum_{p \in \mathcal{Q}(p')} \bar{f}_p \leq \sum_{p \in \mathcal{P}_t, e \in p} \bar{f}_p,$$

since the sets $\mathcal{Q}(p')$ are disjoint. To conclude, since (x, f) is a feasible LP solution,

$$\sum_{p \in \mathcal{P}_t, e \in p} \bar{f}_p = (1 + \frac{1}{\log|S|}) \sum_{p \in \mathcal{P}_t, e \in p} f_p \leq (1 + \frac{1}{\log|S|}) x_e = x_e^i. \qquad \blacktriangleleft$$

The "prune and separate" portion of h-frac-DST is the same as in h-DST (Algorithm 2). Therefore, Claim 9 holds for h-frac-DST. It is not difficult to see that the resulting solution $F = \text{h-frac-DST}(I, (x, f))$ contains an s-t path of length at most $O(\log |S|) \cdot h$ for each terminal $t \in S$, using reasoning analogous to that of Lemma 10. We bound the approximation ratio in the following lemma, which concludes the proof of Theorem 3.

▶ **Lemma 13.** *If (x, f) is a feasible LP solution to the LC-DST instance I, then the cost of the solution returned by* h-frac-DST$(I, (x, f))$ *is at most $O(\log^2 k) \sum_e c(e)x_e$.*

Proof. Let F be the solution returned by h-frac-DST$(I, (x, f))$. We let $X = \sum_e c(e)x_e$. We prove by induction on the number of terminals that $c(F) \leq 12(\log |S| + 1)^2 X$.

In the base case, it is easy to see that the cost of the min-cost h-length-bounded s-t path is at most X. For the recursive step, for each $i \in [\tau]$, let X_i denote the value of the LP solution (x^i, f^i), and let F_i denote the output of h-frac-DST$(I_i, (x^i, f^i))$. By Claim 9, $c(P) \leq 6\gamma = 12X \log |S|$. By induction,

$$c(F) = c(P) + \sum_{i \in [\tau]} c(F_i) \leq 12X \log |S| + \sum_{i \in [\tau]} 12X_i (\log |S \cap C_i| + 1)^2.$$

Since the number of terminals is halved at each recursive step, $\log |S \cap C_i| \leq \log(|S|/2) \leq \log |S| - 1$. Furthermore, since $C_1, \ldots, C_\tau$ are disjoint, $\sum_{i \in [\tau]} X_i \leq (1 + \frac{1}{\log |S|})X$. Thus

$$\begin{aligned} c(F) &\leq 12 \log |S| X + \sum_{i \in [\tau]} 12 (\log |S|)^2 X_i \leq 12 \log |S| X + 12 (\log |S|)^2 (1 + \frac{1}{\log |S|}) X \\ &= 12X(\log^2 |S| + 2 \log |S|) \leq 12X(\log |S| + 1)^2. \end{aligned}$$

This concludes the inductive argument; thus, $c(F) \leq O(\log^2 k) \sum_{e \in E} c(e)x_e$. ◀

4 Length-Constrained Directed Steiner Forest

In this section, we consider length-constrained Directed Steiner Forest in planar digraphs (called length-constrained planar DSF). We are given as input a directed graph $G = (V, E)$ with non-negative edge costs $c(e)$, lengths $\ell(e)$, length constraint h, and demand pairs $D = \{(s_i, t_i)\}_{i \in [k]}$. The goal is to find a min-cost subgraph $H \subseteq G$ such that for each $(s_i, t_i) \in D$, H contains an s_i-t_i path of length at most h. We employ the *junction tree* approach, discussed in Section 1. We formally define an h-length-constrained junction tree.

▶ **Definition 14.** *Given an instance (G, D, h) of LC-DSF, an* h-length-constrained junction tree *that covers terminal pairs $D_H \subseteq D$ is a subgraph $H \subseteq G$ with a root $v \in V(H)$ such that for each $(s_i, t_i) \in D_H$, H contains an s_i-v path and a v-t_i path, each of length at most h. The* density *of H is $c(H)/|D_H|$.*

We first show, in Section 4.1, that given any instance to planar LC-DSF, there exists a low-density length-constrained junction tree. In Section 4.2, we describe the approximation algorithm to find such a junction tree, and conclude the proof of Theorem 4.

4.1 Existence of a good junction tree

We prove the following theorem.

▶ **Theorem 15.** *Given an instance (G, D, h) of length-constrained planar DSF, there exists a $(3h)$-length-constrained junction tree of density at most $O(\log^2 k) \cdot \text{OPT}/k$.*

For planar-DSF *without* length-constraints, [14] showed that every instance of planar-DSF contains a low density junction tree. We outline the approach of [14], as we follow a similar framework. The proof proceeds by considering an optimum solution $E^* \subseteq E$ and finding several junction trees in E^* that are essentially disjoint and cover a large fraction of terminal pairs. The construction of these junction trees follows three steps. The first is a "layering" of a directed graph, where each layer has a tree-like structure, and all paths in the original graph are contained in at most two consecutive layers. This layering scheme was given by Thorup [58] and also used by Kawarabayashi and Sidiropoulos [45] to obtain improved upper bounds on the multicommodity flow-cut gap in directed planar graphs. The second step is a recursive approach based on the planar separator theorem (Lemma 7). The third step is to show that in one level of recursion, if many s_i-t_i paths pass through the separator, then there exist vertices on the separator that behave as roots of low-density junction trees.

The proof of Theorem 15 follows the same three-step strategy. First, we reduce to the case where the optimal solution is an *h-length-constrained* 3*-layered digraph* (Lemma 17).

▶ **Definition 16.** *An h-length-constrained* 3*-layered spanning tree T of a digraph G is a spanning tree of the* undirected version *of G, such that any root-to-leaf path in T is the concatenation of at most* 3 *dipaths in G, each of length at most h. An h-length-constrained* 3*-layered digraph is one that has an h-length-constrained* 3*-layered spanning tree. The root of the digraph is the root of the spanning tree.*

The argument uses a similar decomposition as in [14], but requires additional care to handle lengths. Second, we apply the planar separator and corresponding divide-and-conquer argument to reduce to a so-called "one-path" setting (Lemma 20). This step is essentially identical to the corresponding argument in [14]. The third step is a new contribution: handling the one-path setting under length constraints. The argument of [14] no longer applies, and we give a new argument (Lemma 23).

4.1.1 Reduction to length-constrained layered digraphs

First, we reduce to the case where the optimal solution is an h-length-constrained 3-layered digraph, by proving the following lemma.

▶ **Lemma 17.** *Let (G, D, h) be an instance of length-constrained planar DSF and let $G^* = (V^*, E^*)$ be an optimal solution. Then, there exists a subgraph $H \subseteq G^*$ and a subset of terminals $D_H \subseteq D$ such that the following holds.*

1. *H is an h-length-constrained* 3*-layered digraph with root v;*
2. *For each $(s_i, t_i) \in D_H$, there exists an s_i-t_i path in $H \setminus \{v\}$ of length at most h;*
3. *H is a minor of G^*, and $H \setminus \{v\}$ is a subgraph of G^*.*
4. $\frac{c(E(H))}{|D_H|} \leq 3 \cdot \frac{c(E^*)}{|D|}$.

To prove Lemma 17, we will decompose G^* into a series of 3-layered digraphs and show that at least one satisfies the necessary properties. We assume G^* is weakly connected; else, this decomposition can be applied separately on each weakly connected component. Let v_0 be an arbitrary node in V^*. Let L_0 be the set of all nodes in V^* that are reachable from v_0 (in G^*) with a path of length at most h. We then define alternating layers until all vertices have been covered by a layer.

$$L_j = \begin{cases} \{u \in V^* \setminus \cup_{j'<j} L_{j'} : \ell(u, L_{j-1}) \leq h\} & j \text{ is odd} \\ \{u \in V^* \setminus \cup_{j'<j} L_{j'} : \ell(L_{j-1}, u) \leq h\} & j \text{ is even} \end{cases}$$

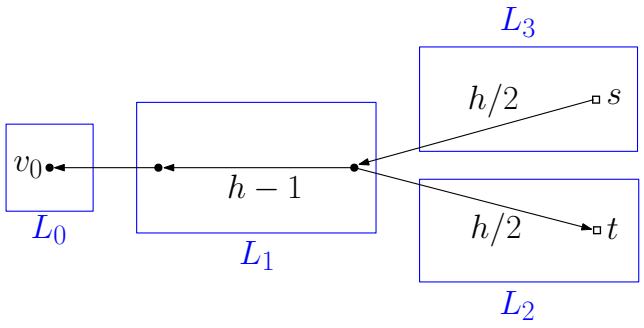


Figure 4 Example of graph in which s-t path spans 3 layers, L_1, L_2, L_3.

Let q denote the total number of layers. For each layer $j \leq q-2$, we define G_j to be the graph obtained from G^* by contracting $\cup_{j'<j} L_{j'}$ into a new node v_j, and deleting $\cup_{j''>j+2} L_{j''}$.

▷ Claim 18. For each demand pair $(s_i, t_i) \in D$, there exists $j \in \{0, \ldots, q-2\}$ such that $G_j \setminus \{v_j\}$ contains an s_i-t_i path of length at most h.

Proof. Let P_i be an s_i-t_i path of length at most h in G^*; such a path must exist as G^* is a feasible solution to the given LC-DSF instance. Let j be the minimum index such that $P_i \cap L_j \neq \emptyset$. We assume j is odd; the case where j is even is similar. Let u be in $P_i \cap L_j$. Then, since $\ell(u, t_i) \leq h$, t_i must be contained in $\cup_{j' \leq j+1} L_{j'}$. Since all of P_i can reach t_i with a path of length at most h, $P_i \subseteq \cup_{j' \leq j+2} L_{j'}$. Therefore, since L_j is the first layer that intersects with P_i, $P_i \subseteq L_j \cup L_{j+1} \cup L_{j+2} \subseteq G_j \setminus \{v_j\}$.[5] ◁

▶ Remark 19. When considering planar-DSF *without* length constraints, [14] shows that all s_i-t_i paths are contained in at most 2 layers. With length constraints, however, 3 layers are needed; see Figure 4 for an example of a graph in which the s-t path spans 3 layers.

Proof of Lemma 17. For each $j \in \{0, \ldots, q-2\}$, let D_j denote the set of terminal pairs for which $G_j \setminus v_j$ contains an s_i-t_i path of length at most h. We show that the first three properties required by Lemma 17 hold for all (G_j, D_j). The first and third properties follow by construction: each G_j is an h-length-constrained 3-layered digraph with root v_j, and G_j is constructed from G^* via contraction (to form the root) and deletion. The second property follows by definition of D_j.

To conclude, we show that there exists some (G_j, D_j) that satisfies the fourth property. By Claim 18, $\cup_j D_j = D$. Furthermore, it is not difficult to see that $\sum_{j<q-1} c(G_j) \leq 3c(E^*)$, since each edge of E^* is either contained in one layer L_j, or goes between two layers L_j and L_{j+1} – in either case, it exists only in G_j, G_{j-1}, and G_{j-2}. By an averaging argument, there must be some $j \in [q-2]$ such that

$$\frac{c(G_j)}{|D_j|} \leq \frac{\sum_j c(G_j)}{\sum_j |D_j|} \leq \frac{3c(E^*)}{|D|}.$$ ◀

[5] We point out a minor technicality – this claim is not true when $j = 0$. However, one can circumvent this issue by creating a new dummy vertex as the root of G_0 and adding a 0-length edge to v_0.

4.1.2 Divide-and-conquer via planar separator

Next, we use the fact that planar digraphs have good separators to reduce to the so-called "one-path" case. We prove the following lemma.

▶ **Lemma 20.** *Let $G^* = (V^*, E^*)$ be a planar h-length-constrained 3-layered digraph with root v, and let $D \subseteq V^* \times V^*$ be a set of demand pairs such that for each $(s_i, t_i) \in D$, $G^* \setminus \{v\}$ has an s_i-t_i path of length at most h. Then, there exists a subgraph $H \subseteq G^* \setminus \{v\}$, an h-length path $P \subseteq H$, and subset of terminals $D_H \subseteq D$ satisfying the following.*

1. *For each $(s_i, t_i) \in D_H$, there exists an h-length s_i-t_i path in H that intersects P;*
2. $\frac{c(E(H))}{|D_H|} \leq O(\log |D|) \frac{c(E^*)}{|D|}$.

The proof follows the same argument as that of [14]; we rewrite it here for completeness and borrow the terminology. Fix an instance (G, D) of planar-DSF and a feasible solution E^* satisfying the conditions outlined in the statement of Lemma 20 above. Let $T^* \subseteq E^*$ be an h-length-constrained 3-layered spanning tree of G^*. We will follow a recursive process to partition D into subsets on which we build junction trees.

- Set vertex weights as $w(s_i) = w(t_i) = 1$ for all $(s_i, t_i) \in D$, and 0 otherwise.
- Consider the undirected version of E^* and apply Lemma 7 on the undirected version of spanning tree T^* with vertex weights w. From this, we obtain u_1, u_2, u_3.
- Since T^* is an h-length-constrained 3-layered tree, each (undirected) path $P_{T^*}(v, u_i)$ consists of at most 3 dipaths, each of length at most h. Remove the root v and let Q_i^j for $i, j \leq 3$ denote the 3 dipaths of $P_{T^*}(v, u_i) \setminus \{v\}$. Let $S_0 = \cup_{i,j \leq 3} Q_i^j$ denote this collection of at most 9 dipaths, we call this the *separator*.
- Define $D_0 \subseteq D$ to be the set of all terminal pairs (s_i, t_i) such that E^* contains an h-length s_i-t_i path that intersects one of the dipaths in S_0. Equivalently, $(s_i, t_i) \in D_0$ iff there exists an h-length s_i-t_i path $P_i \subseteq E^*$ such that $V(P_i) \cap V(S_0) \neq \emptyset$. We say these terminal pairs are *separated*.
- Let $\mathcal{C}_1$ be the set of weakly connected components of $G \setminus (\cup_{i \in [3]} P_{T^*}(v, u_i))$; we drop "weakly connected" and simply refer to these as "components" in the remainder of this section. Note that each $C \in \mathcal{C}_1$ has at most half the total number of terminals.

We then follow a recursive process, similar to PruneAndSeparate described in Section 3. For each $C \in \mathcal{C}_1$, we contract S_0 into v and recurse on the sub-instance consisting of $G^*[C \cup \{v\}]$ with terminal set $\{(s_i, t_i) \in D : s_i, t_i \in C\} \setminus D_0$. It is easy to verify that $G^*[C \cup \{v\}]$ is an h-length-constrained 3-layered digraph with root v. Furthermore, each (s_i, t_i) pair in the new terminal set has an h-length-bounded path that does not intersect v; else, (s_i, t_i) would have been included in D_0. Therefore, we can apply the same process as above, and repeat this recursive process until each component has at most one terminal. Since the number of terminals halves at each step, the recursion depth is at most $\lceil \log 2k \rceil = \lceil \log k \rceil + 1$.

We use the following notation. For an instance defined on component C at recursive depth j, we let S_j^C, D_j^C denote the separator and corresponding separated terminal pairs. We let $\mathcal{C}_j$ denote the set of all components at recursive depth j (in particular, $\mathcal{C}_0 = \{G^*\}$), and denote by $S_j := \cup_{C \in \mathcal{C}_j} S_j^C$ and $D_j := \cup_{C \in \mathcal{C}_j} D_j^C$. The following claim is simple; the proof is deferred to a full version.

▷ Claim 21. $D \subseteq \cup_{j=0}^{\lceil \log k \rceil + 1} D_j$.

▶ **Corollary 22.** *There exists a level of recursion $j^* \in \{0, \ldots, \lceil \log k \rceil + 1\}$ such that $|D_{j^*}| \geq \frac{k}{\lceil \log k \rceil + 2}$.*

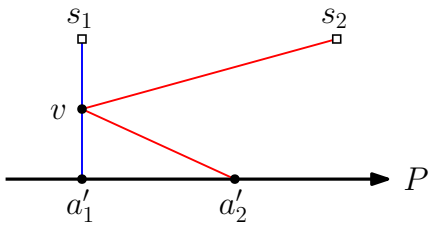


Figure 5 Suppose $\ell(s_1, v) = 1, \ell(v, a'_1) = h - 1, \ell(s_2, v) = h - 1, \ell(v, a'_2) = 1$. The argument in [14] would re-route s_2 to P via the path from v to a'_1; however, this would increase the length of the path from s_2 to P to $2h - 2$. In general, merging arbitrarily many terminals would make these paths to P arbitrarily long.

Proof of Lemma 20. Let j^* be the recursion level given by Corollary 22 such that $|D_{j^*}| \geq \frac{k}{\lceil \log k \rceil + 2}$. Since components of $\mathcal{C}_{j^*}$ are disjoint, $\sum_{C \in \mathcal{C}_{j^*}} c(E(C)) \leq c(E^*)$ and $\{D^C_{j^*}\}_{C \in \mathcal{C}_{j^*}}$ form a partition of D_{j^*}. Using an averaging argument, there exists a component C such that $c(E(C))/|D^C_{j^*}| \leq c(E^*)/|D_{j^*}|$. Furthermore, since $S^C_{j^*}$ consists of at most 9 dipaths, there exists at least one dipath Q^* such that at least $\frac{1}{9}$ terminal pairs of $D^C_{j^*}$ intersect Q^*; we denote this set of terminals by D^*.

We claim C, Q^*, D^* satisfy the required properties of the lemma statement. The first follows by construction; for all terminal pairs $(s_i, t_i) \in D^*$, there exists an h-length s_i-t_i path in C that intersects Q^*. For the second, $\frac{c(E(C))}{|D^*|} \leq 9 \cdot \frac{c(E(C))}{|D^C_{j^*}|} \leq 9 \cdot \frac{c(E^*)}{|D_{j^*}|} \leq O(\log k) \frac{c(E^*)}{|D|}$. ◀

4.1.3 One Path Case

To conclude, we show that in the "one-path" setting, there always exists a low-density junction tree. This subsection contains the core technical contribution over [14], since previously used techniques for handling the one-path setting do not extend to length-constraints.

▶ **Lemma 23.** *Let $G^* = (V^*, E^*)$ be a planar subgraph that contains an h-length path $P \subseteq G^*$, and let $D \subseteq V^* \times V^*$ be a set of demand pairs such that for each $(s_i, t_i) \in D$, there exists an h-length s_i-t_i path in G^* that intersects P. Then, G^* contains a $3h$-length-constrained junction tree of density $O(\log k)c(E^*)/k$, where $k = |D|$.*

Let $v_1, \ldots, v_{|P|}$ denote the vertices of P. For each terminal pair $(s_i, t_i) \in D$, the lemma assumption guarantees an h-length s_i-t_i path in G^* that intersects P. Let a'_i denote the first vertex of P on this path, and let b'_i denote the last vertex of P on this path; note that $a'_i \leq_P b'_i$. Let P_{s_i} be the subpath from s_i to a'_i, and let P_{t_i} be the subpath from b'_i to t_i; both have length at most h. We can assume without loss of generality that $E^* = P \cup (\cup_{i \in [k]}(P_{s_i} \cup P_{t_i}))$, since this satisfies the lemma assumptions. The main challenge with length constraints in the one-path setting is in handling edges of $E^* \setminus P$. In particular, the argument in [14] relies on the fact that for any $i, i' \in [k]$, if $P_{s_i} \cap P_{s_{i'}} \neq \emptyset$, then $a'_i = a'_{i'}$. This no longer holds, as demonstrated by Figure 5.

We adopt a different perspective. For each $(s_i, t_i) \in D$, let a_i denote the first node in P that s_i can reach within a path of length *at most* $2h$, and let b_i denote the last node in P that can reach t_i within length at most $2h$. We use the following simple claim.

▷ Claim 24. For all $i \in [k]$, $a_i \leq_P a'_i \leq_P b'_i \leq_P b_i$.

Proof. By definition, $a'_i \leq_P b'_i$. For the other two inequalities, $a_i \leq_P a'_i$, since a_i is the *first* node on P that s_i can reach within length $2h$, and P_{s_i} is an s-P path of length h. Similarly, $b'_i \leq_P b_i$, since b_i is the *last* node on P that can reach t_i within length $2h$, and P_{t_i} is a path from P to t_i of length h. ◁

Let μ be the midpoint of the path P (i.e., $\mu = v_{\lfloor |P|/2 \rfloor}$). We partition D into three sets:

$$\begin{cases} L = \{(s_i, t_i) : a_i \leq_P b_i <_P \mu\} \\ R = \{(s_i, t_i) : b_i \geq_P a_i >_P \mu\} \\ M = \{(s_i, t_i) : a_i \leq_P \mu \leq_P b_i\} \end{cases}$$

▷ Claim 25. E^* contains a $3h$-length-constrained junction tree with root μ on terminals M.

Proof. It suffices to show that for each $(s_i, t_i) \in M$, there exist $3h$-length paths in E^* from s_i to μ and from μ to t_i. By definition, $\ell(s_i, a_i) \leq 2h$. Furthermore, since $a_i \leq_P \mu$ and P is an h-length path, $\ell(a_i, \mu) \leq h$. Therefore, $\ell(s_i, \mu) \leq 3h$. The argument for t_i is similar. ◁

We follow a recursive process. First, we construct the junction tree H_μ with root μ given by Claim 25. We conflate indices with vertices, and let $(\mu - 1)$ and $(\mu + 1)$ denote the vertices directly before and after μ on P. At a high level, the idea is then to recurse on the sub-paths $P[0, \mu - 1]$, $P[\mu + 1, v_{|P|}]$, with corresponding paths in $E \setminus P$ to connect terminals. Formally, let $E_L = P[0, \mu - 1] \cup \big(\cup_{(s_i,t_i) \in L}(P_{s_i} \cup P_{t_i})\big)$, and $E_R = P[\mu + 1, v_{|P|}] \cup \big(\cup_{(s_i,t_i) \in R}(P_{s_i} \cup P_{t_i})\big)$. We recurse on the edge-induced subgraph $G_L := G[E_L]$ with terminal set L, and $G_R := G[E_R]$ with terminal set R. The following claim follows directly from Claim 24, and proves that these are valid recursive calls.

▷ Claim 26. For each $(s_i, t_i) \in D$, recall that a'_i, b'_i denote the endpoints on P of P_{s_i}, P_{t_i} respectively. If $(s_i, t_i) \in L$, then $a'_i, b'_i \in P[0, \mu - 1]$, and if $(s_i, t_i) \in R$, then $a'_i, b'_i \in P[\mu + 1, v_{|P|}]$.

We continue this recursive process until P consists of one vertex v, in which case all remaining terminal pairs can be routed through v. This process terminates after $O(\log |P|) = O(\log k)$ recursive steps, since $|P|$ is halved at each step. The following key lemma bounds the total cost of the junction trees constructed. Let $E_s = E(\cup_{i \in [k]} P_{s_i})$ and $E_t = E(\cup_{i \in [k]} P_{t_i})$.

▷ Claim 27. $E_s \cap E_L$ is disjoint from $E_s \cap E_R$. Similarly, $E_t \cap E_L$ is disjoint from $E_t \cap E_R$.

Proof. We prove the claim for E_s; the argument for E_t is similar. Suppose for the sake of contradiction there exists an edge $e = (u, v)$ in $E_s \cap E_L \cap E_R$. Since $e \in E_L \cap E_s$, there must be a terminal s_i such that $e \in P_{s_i}$ and $a_i \leq_P b_i <_P \mu$. By Claim 24, $a'_i <_P \mu$. Similarly, since $e \in E_R$, there exists s_j such that $e \in P_{s_j}$ and $a_j >_P \mu$. Consider the path $P' = P_{s_j}[s_j, u] \circ P_{s_i}[u, a'_i]$. Since P_{s_j} and P_{s_i} are both h-length paths, P' has length at most $2h$. Thus, s_j can reach a'_i with a path of length $\leq 2h$, and $a'_i <_P a_j$, a contradiction. ◁

Proof of Lemma 23. The recursive process described yields a series of junction trees denoted H_v for some vertices $v \in P$ that, in total, cover all terminal pairs. By Claim 25, each H_v is a $3h$-length-constrained junction tree. To bound the cost, it is helpful to think of E_s, E_t and $E(P)$ as disjoint sets; this can be achieved by making at most 3 copies of each edge in E^*. Then, by Claim 27, E_L is disjoint from E_R. Therefore, each edge is in at most one junction tree per recursive level, so it is in at most $O(\log k)$ total junction trees. Therefore, the total cost of all junction trees is at most $3 \cdot O(\log k) c(E^*) = O(\log k) c(E^*)$. By an averaging argument, there must be one junction tree of density at most $O(\log k) c(E^*)/k$. ◀

To conclude, the proof of Theorem 15 follows from Lemma 17, Lemma 20, and Lemma 23.

4.2 Finding an Approximate Low-Density Junction Tree

The main lemma we prove in this section is the following.

▶ **Lemma 28.** *Suppose there exists a bicriteria (α, β)-approximation for LC-DST in planar graphs with respect to the optimal solution to LC-DST-LP. Then, given a planar-LC-DSF instance (G, D, h), there exists an efficient algorithm to obtain a βh-length-constrained junction tree of G of density at most $O(\alpha \cdot \log k)$ times the optimal.*

The algorithm is as follows. First, we guess the root of the junction tree; we can run this algorithm for each candidate root $v \in V$ and return the junction tree with minimum density. Next, we solve the following LP relaxation for the min-density junction tree rooted at v. We follow a similar structure to that of LC-DST-LP, with additional variables y_i for each $i \in [k]$ to indicate whether or not the pair (s_i, t_i) is covered in the solution. For each $(s_i, t_i) \in D$, we let $\mathcal{P}^h_{s_i}$ denote the set of all h-length-constrained paths from s_i to v, and $\mathcal{P}^h_{t_i}$ denote the set of all h-length-constrained paths from v to t_i. Following LC-DST-LP, we have variables x_e for each $e \in E$ indicating whether an edge e is in junction tree, and variables f_p indicating which paths are used to connect s_i to v and v to t_i. The resulting density of the junction tree would be $(\sum_{e \in E} c(e)x_e)/(\sum_{i \in [k]} y_i)$; we normalize $\sum_{i \in [k]} y_i = 1$.

$$
\begin{aligned}
\min \quad & \sum_{e \in E} c(e) x_e & & \\
s.t. \quad & \sum_{p \in \mathcal{P}^h_{s_i}} f_p \geq y_i & & \forall i \in [k] \\
& \sum_{p \in \mathcal{P}^h_{t_i}} f_p \geq y_i & & \forall i \in [k] \\
& \sum_{p \in \mathcal{P}^h_{s_i}, e \in p} f_p \leq x_e & & \forall e \in E, i \in [k] \\
& \sum_{p \in \mathcal{P}^h_{t_i}, e \in p} f_p \leq x_e & & \forall e \in E, i \in [k] \\
& \sum_{i \in [k]} y_i = 1 & & \\
& x_e, f_p, y_i \geq 0 & & \forall e \in E, p \in \cup_{i \in [k]} \left(\mathcal{P}^h_{s_i} \cup \mathcal{P}^h_{t_i} \right), i \in [k]
\end{aligned}
\tag{LC-Den-LP}
$$

Let x^*, f^*, y^* be an optimal solution to LC-Den-LP. Partition the set of terminal pairs into groups $D_0, \ldots, D_{\log k}$, where $(s_i, t_i) \in D_j$ if $y^*_i \in (\frac{1}{2^{j+1}}, \frac{1}{2^j}]$. We will abuse notation and sometimes write $i \in D_j$ for ease.

▷ Claim 29. There exists $\theta \in \{0, \ldots, \log k\}$ such that $\sum_{i \in D_\theta} y^*_i \geq 1/(2(\log k + 1))$.

Proof. If $(s_i, t_i) \in D \setminus (\cup_{j=0}^{\log k} D_j)$, then $y^*_i \leq \frac{1}{2^{\log k + 1}} = \frac{1}{2k}$. Thus, $\sum_{i \notin \cup_{j=0}^{\log k} D_j} y^*_i \leq \sum_{i \notin \cup_{j=0}^{\log k} D_j} \frac{1}{2k} \leq \frac{1}{2}$. Recall that $\sum_{i \in [k]} y^*_i = 1$. Therefore, $\sum_{i \in \cup_{j=0}^{\log k} D_j} y^*_i \geq \frac{1}{2}$. Since there are $\log k + 1$ disjoint groups, there must be at least one group D_θ with $\sum_{i \in D_\theta} y^*_i$ at least $1/(2(\log k + 1))$. ◁

Let θ be given by Claim 29. We use the (α, β)-approximation algorithm for LC-DST twice: first, we consider the instance on G with root v and terminal set $D^t_\theta = \{t_i : (s_i, t_i) \in D_\theta\}$, and obtain a solution T_t. Second, we let G^R be obtained from G by reversing the direction of all edges. We consider the LC-DST instance on G^R with root v and terminal set

$D^s_\theta = \{s_i : (s_i, t_i) \in D_\theta\}$ and obtain a solution T'_s in G^R. Note that T'_s contains an h-length-constrained v-s_i path in G^R for all $s_i \in D^s_\theta$. Thus, if we reverse all edges of T'_s, the resulting graph T_s is a subgraph of G containing an h-length-constrained s_i-v path for all $s_i \in D^s_\theta$. Therefore, $T = T_t \cup T_s$ is a (βh)-length-constrained junction tree on terminal pairs D_θ.

▷ Claim 30. Let $\lambda = 2^{\theta+1}$. Then $(\lambda x^*, \lambda f^*)$ is a feasible solution to LC-DST-LP on both of the following instances:

- G with terminal set D^t_θ,
- G^R with terminal set D^s_θ.

In both cases, f^* is restricted to relevant paths for terminals in D^t_θ, D^s_θ.

Proof. We prove the claim for G^R with terminal set D^s_θ, as the argument for (G, D^t_θ) is analogous. Recall that $\mathcal{P}^h_{s_i}$ is the set of all h-length-constrained s_i-v paths in G. Thus, each path in $\mathcal{P}^h_{s_i}$ is an h-length constrained v-s_i path in G^R. Since (x^*, f^*, y^*) is a feasible solution to LC-Den-LP, $\sum_{p\in\mathcal{P}^h_{s_i}} f^*_p \geq y^*_i$. Furthermore, since $i \in D_\theta$, $y^*_i > \frac{1}{2^{\theta+1}}$. Thus, $\sum_{p\in\mathcal{P}^h_{s_i}} \lambda f^*_p \geq \lambda y^*_i > 1$.

The fact that $(\lambda x^*, \lambda f^*)$ satisfies the second constraint of LC-DST-LP follows immediately from the fact that (x^*, f^*) satisfies the third constraint of LC-DST-LP, since both x^* and f^* are scaled by the same value. ◁

▷ Claim 31. The density of T is at most $O(\alpha \log k) \sum_{e\in E} c(e)x^*_e$.

Proof. The algorithms used to construct T_s and T_t are (α, β)-approximations with respect to the LP. This, along with Claim 30, implies that the costs of T_t and T_s are each upper bounded by $\alpha\lambda \sum_{e\in E} c(e)x^*_e$, where $\lambda = 2^{\theta+1}$.

To bound $|D_\theta|$, observe that $\sum_{i\in D_\theta} y^*_i \leq \sum_{i\in D_\theta} 1/2^\theta = |D_\theta|/2^\theta$. Furthermore, recall that θ was defined by Claim 29 such that $\sum_{i\in D_\theta} y^*_i \geq 1/(2(\log k + 1))$. Therefore, $|D_\theta| \geq 2^\theta \sum_{i\in D_\theta} y^*_i \geq \frac{2^\theta}{2(\log k+1)}$. The density of T is at most

$$\frac{c(T_t) + c(T_s)}{|D_\theta|} \leq \frac{2\log k + 2}{2^\theta} \left(2\alpha\lambda \sum_{e\in E} c(e)x^*_e\right) = 8\alpha(\log k + 1) \sum_{e\in E} c(e)x^*_e. \quad \triangleleft$$

Lemma 28 follows from Claim 31 and the fact that T is a (βh)-length-constrained junction tree. Combining Lemma 28 with the bicriteria LP-competitive approximation for LC-DST in planar digraphs (Theorem 3) then yields Theorem 32.

▶ **Theorem 32.** *Given an instance (G, D, h) of LC-DSF in planar digraphs, there exists an efficient algorithm to obtain an $O(\log k)$-length-constrained junction tree of G of density at most $O(\log^3 k)$ times the optimal.*

This, along with Theorem 15, yields an efficient algorithm to find an $O(\log k)$-length-constrained junction tree of density at most $O(\log^5 k)\mathrm{OPT}/k$. By the standard covering argument, this yields a bicriteria $(O(\log^6 k), O(\log k))$-approximation for LC-DSF in planar digraphs, concluding the proof of Theorem 4.

References

1 Amir Abboud and Greg Bodwin. Reachability preservers: New extremal bounds and approximation algorithms. *SIAM Journal on Computing*, 53(2):221–246, 2024. doi:10.1137/21M1442176.

2 Ibrahim Akgün and Barbaros Ç Tansel. New formulations of the hop-constrained minimum spanning tree problem via Miller-Tucker-Zemlin constraints. *European Journal of Operational Research*, 212(2):263–276, 2011. doi:10.1016/J.EJOR.2011.01.051.

3 Ernst Althaus, Stefan Funke, Sariel Har-Peled, Jochen Könemann, Edgar A. Ramos, and Martin Skutella. Approximating k-hop minimum-spanning trees. *Operations Research Letters*, 33(2):115–120, 2005. doi:10.1016/j.orl.2004.05.005.

4 Spyridon Antonakopoulos, Chandra Chekuri, Bruce Shepherd, and Lisa Zhang. Buy-at-bulk network design with protection. *Mathematics of Operations Research*, 36(1):71–87, 2011. doi:10.1287/moor.1110.0484.

5 Anantaram Balakrishnan and Kemal Altinkemer. Using a hop-constrained model to generate alternative communication network design. *ORSA Journal on Computing*, 4(2):192–205, 1992. doi:10.1287/ijoc.4.2.192.

6 M. Bateni, M. Hajiaghayi, and D. Marx. Approximation schemes for steiner forest on planar graphs and graphs of bounded treewidth. *Journal of the ACM (JACM)*, 58(5):21, 2011.

7 Piotr Berman, Arnab Bhattacharyya, Konstantin Makarychev, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Approximation algorithms for spanner problems and directed Steiner forest. *Information and Computation*, 222:93–107, 2013. 38th International Colloquium on Automata, Languages and Programming (ICALP 2011). doi:10.1016/j.ic.2012.10.007.

8 Aaron Bernstein. Near linear time $(1+\epsilon)$-approximation for restricted shortest paths in undirected graphs. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 189–201. SIAM, 2012. doi:10.1137/1.9781611973099.17.

9 Glencora Borradaile, Philip Klein, and Claire Mathieu. An $O(n \log n)$ approximation scheme for Steiner tree in planar graphs. *ACM Transactions on Algorithms (TALG)*, 5(3):1–31, 2009. doi:10.1145/1541885.1541892.

10 Moses Charikar, Chandra Chekuri, To-Yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed Steiner problems. *Journal of Algorithms*, 33(1):73–91, 1999. doi:10.1006/JAGM.1999.1042.

11 C. Chekuri, M. T. Hajiaghayi, G. Kortsarz, and M. R. Salavatipour. Approximation algorithms for nonuniform buy-at-bulk network design. *SIAM Journal on Computing*, 39(5):1772–1798, 2010. doi:10.1137/090750317.

12 Chandra Chekuri, Guy Even, Anupam Gupta, and Danny Segev. Set connectivity problems in undirected graphs and the directed steiner network problem. *ACM Transactions on Algorithms (TALG)*, 7(2):1–17, 2011. doi:10.1145/1921659.1921664.

13 Chandra Chekuri and Rhea Jain. Approximation algorithms for hop constrained and buy-at-bulk network design via hop constrained oblivious routing. In Timothy Chan, Johannes Fischer, John Iacono, and Grzegorz Herman, editors, *32nd Annual European Symposium on Algorithms (ESA 2024)*, volume 308 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 41:1–41:21, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ESA.2024.41.

14 Chandra Chekuri and Rhea Jain. A polylogarithmic approximation for Directed Steiner Forest in planar digraphs. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2095–2110, 2025. doi:10.1137/1.9781611978322.67.

15 Chandra Chekuri and Rhea Jain. Node-weighted multicut in planar digraphs. *arXiv preprint*, 2026. doi:10.48550/arXiv.2601.20038.

16 Chandra Chekuri and Rhea Jain. A polylogarithmic approximation for buy-at-bulk network design with protection. In *Proceedings of the Fifty-Eighth Annual ACM Symposium on Theory of Computing (STOC)*, 2026. to be published.

17 Chandra Chekuri, Rhea Jain, Shubhang Kulkarni, Da Wei Zheng, and Weihao Zhu. From Directed Steiner Tree to Directed Polymatroid Steiner Tree in Planar Graphs. In Timothy Chan, Johannes Fischer, John Iacono, and Grzegorz Herman, editors, *32nd Annual European Symposium on Algorithms (ESA 2024)*, volume 308 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 42:1–42:19, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ESA.2024.42.

18 Chandra Chekuri and Martin Pal. A recursive greedy algorithm for walks in directed graphs. In *46th annual IEEE symposium on foundations of computer science (FOCS'05)*, pages 245–253. IEEE, 2005.

19 Eden Chlamtáč, Michael Dinitz, Guy Kortsarz, and Bundit Laekhanukit. Approximating spanners and directed steiner forest: Upper and lower bounds. *ACM Transactions on Algorithms (TALG)*, 16(3):1–31, 2020. doi:10.1145/3381451.

20 Vincent Cohen-Addad. Bypassing the surface embedding: approximation schemes for network design in minor-free graphs. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 343–356, New York, NY, USA, June 2022. Association for Computing Machinery. doi:10.1145/3519935.3520049.

21 Geir Dahl. The 2-hop spanning tree problem. *Operations Research Letters*, 23(1):21–26, 1998. doi:10.1016/S0167-6377(98)00029-7.

22 Erik D Demaine, MohammadTaghi Hajiaghayi, and Philip N Klein. Node-weighted Steiner tree and group Steiner tree in planar graphs. *ACM Transactions on Algorithms (TALG)*, 10(3):1–20, 2014. doi:10.1145/2601070.

23 Michael Dinitz, Guy Kortsarz, and Ran Raz. Label cover instances with large girth and the hardness of approximating basic k-spanner. *ACM Trans. Algorithms*, 12(2), December 2016. doi:10.1145/2818375.

24 Yevgeniy Dodis and Sanjeev Khanna. Design networks with bounded pairwise distance. In *Proceedings of the thirty-first annual ACM Symposium on Theory of Computing (STOC)*, pages 750–759, 1999. doi:10.1145/301250.301447.

25 Moran Feldman, Guy Kortsarz, and Zeev Nutov. Improved approximation algorithms for Directed Steiner Forest. *Journal of Computer and System Sciences*, 78(1):279–292, January 2012. doi:10.1016/j.jcss.2011.05.009.

26 A. Filtser. Hop-constrained metric embeddings and their applications. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 492–503, Los Alamitos, CA, USA, February 2022. IEEE Computer Society. doi:10.1109/FOCS52979.2021.00056.

27 Zachary Friggstad, Jochen Könemann, Young Kun-Ko, Anand Louis, Mohammad Shadravan, and Madhur Tulsiani. Linear programming hierarchies suffice for directed Steiner tree. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 285–296. Springer, 2014. doi:10.1007/978-3-319-07557-0_24.

28 Zachary Friggstad and Ramin Mousavi. A constant-factor approximation for quasi-bipartite directed Steiner tree on minor-free graphs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2023)*, pages 13:1–13:18, 2023. doi:10.4230/LIPIcs.APPROX/RANDOM.2023.13.

29 Zachary Friggstad and Ramin Mousavi. A $O(\log k)$-Approximation for Directed Steiner Tree in Planar Graphs. *ACM Transactions on Algorithms*, 21(4):1–14, 2025. Preliminary version in ICALP 2023.

30 Mohsen Ghaffari, Bernhard Haeupler, and Goran Zuzic. Hop-constrained oblivious routing. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, pages 1208–1220, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3406325.3451098.

31 Rohan Ghuge and Viswanath Nagarajan. Quasi-polynomial algorithms for submodular tree orienteering and directed network design problems. *Mathematics of Operations Research*, 47(2):1612–1630, 2022. doi:10.1287/MOOR.2021.1181.

32 Ashish Goel, Kajamalai G Ramakrishnan, Deepak Kataria, and Dimitris Logothetis. Efficient computation of delay-sensitive routes from one source to all destinations. In *Proceedings IEEE INFOCOM*, volume 2, pages 854–858. IEEE, 2001.

33 Fabrizio Grandoni, Bundit Laekhanukit, and Shi Li. $O(\log^2 k/\log\log k)$-approximation algorithm for directed Steiner tree: A tight quasi-polynomial time algorithm. *SIAM Journal on Computing*, 52(2):298–322, 2022. Preliminary version in Proc. of STOC 2019.

34 Elena Grigorescu, Nithish Kumar, and Young-San Lin. Directed Buy-At-Bulk Spanners. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2025)*, volume 353 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.APPROX/RANDOM.2025.22.

35 Bernhard Haeupler, D. Ellis Hershkowitz, and Goran Zuzic. Tree embeddings for hop-constrained network design. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, pages 356–369, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3406325.3451053.

36 Mohammad Taghi Hajiaghayi, Guy Kortsarz, and Mohammad R Salavatipour. Approximating buy-at-bulk and shallow-light k-steiner trees. *Algorithmica*, 53:89–103, 2009. doi:10.1007/S00453-007-9013-X.

37 Eran Halperin and Robert Krauthgamer. Polylogarithmic inapproximability. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 585–594, 2003. doi:10.1145/780542.780628.

38 Refael Hassin. Approximation schemes for the restricted shortest path problem. *Mathematics of Operations research*, 17(1):36–42, 1992. doi:10.1287/MOOR.17.1.36.

39 D Ellis Hershkowitz and Richard Z Huang. Planar length-constrained minimum spanning trees. *arXiv preprint*, 2025. To appear in STOC 2026. doi:10.48550/arXiv.2510.09002.

40 D Ellis Hershkowitz and Richard Z Huang. Simple length-constrained minimum spanning trees. In *2025 Symposium on Simplicity in Algorithms (SOSA)*, pages 341–349, 2025. doi:10.1137/1.9781611978315.25.

41 Rhea Jain. *Approximation Algorithms for Connectivity and Fault-Tolerant Network Design*. University of Illinois at Urbana-Champaign, 2026.

42 David S Johnson and Michael R Garey. *Computers and intractability: A guide to the theory of NP-completeness*. WH Freeman, 1979.

43 Hans C Joksch. The shortest route problem with constraints. *Journal of Mathematical analysis and applications*, 14(2):191–197, 1966.

44 Erez Kantor and David Peleg. Approximate hierarchical facility location and applications to the shallow steiner tree and range assignment problems. In Tiziana Calamoneri, Irene Finocchi, and Giuseppe F. Italiano, editors, *Algorithms and Complexity*, pages 211–222, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. doi:10.1007/11758471_22.

45 Ken-ichi Kawarabayashi and Anastasios Sidiropoulos. Embeddings of planar quasimetrics into directed ℓ_1 and polylogarithmic approximation for directed sparsest-cut. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 480–491. IEEE, 2021. doi:10.1109/FOCS52979.2021.00055.

46 M. Reza Khani and Mohammad R. Salavatipour. Improved approximations for buy-at-bulk and shallow-light k-steiner trees and $(k, 2)$-subgraph. *J. Comb. Optim.*, 31(2):669–685, February 2016. doi:10.1007/s10878-014-9774-5.

47 Jochen Könemann, Asaf Levin, and Amitabh Sinha. Approximating the degree-bounded minimum diameter spanning tree problem. *Algorithmica*, 41:117–129, 2005. doi:10.1007/S00453-004-1121-2.

48 Guy Kortsarz and Zeev Nutov. Approximating some network design problems with node costs. *Theoretical Computer Science*, 412(35):4482–4492, 2011. doi:10.1016/j.tcs.2011.04.013.

49 Guy Kortsarz and David Peleg. Approximating shallow-light trees. In *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '97, pages 103–110, USA, 1997. Society for Industrial and Applied Mathematics.

50 Bundit Laekhanukit. On the integrality gap of directed Steiner tree LPs with relatively integral solutions. *arXiv preprint*, 2024. arXiv:2412.10744.

51 Shi Li and Bundit Laekhanukit. Polynomial integrality gap of flow LP for directed Steiner tree. *ACM Transactions on Algorithms*, 21(1):1–9, 2024. doi:10.1145/3681791.

52 Richard J. Lipton and Robert Endre Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979. doi:10.1137/0136016.

53 Dean H Lorenz and Danny Raz. A simple efficient approximation scheme for the restricted shortest path problem. *Operations Research Letters*, 28(5):213–219, 2001. doi:10.1016/S0167-6377(01)00069-4.

54 Madhav V Marathe, R Ravi, Ravi Sundaram, S.S Ravi, Daniel J Rosenkrantz, and Harry B Hunt. Bicriteria network design problems. *Journal of Algorithms*, 28(1):142–171, 1998. doi:10.1006/jagm.1998.0930.

55 Hasan Pirkul and Samit Soni. New formulations and solution procedures for the hop constrained network design problem. *European Journal of Operational Research*, 148(1):126–140, 2003. doi:10.1016/S0377-2217(02)00366-1.

56 R. Ravi. Rapid rumor ramification: approximating the minimum broadcast time. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 202–213, 1994. doi:10.1109/SFCS.1994.365693.

57 Thomas Rothvoß. Directed Steiner tree and the Lasserre hierarchy. *CoRR*, abs/1111.5473, 2011. arXiv:1111.5473.

58 Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *J. ACM*, 51(6):993–1024, November 2004. doi:10.1145/1039488.1039493.

59 Arthur Warburton. Approximation of pareto optima in multiple-objective, shortest-path problems. *Operations research*, 35(1):70–79, 1987. doi:10.1287/OPRE.35.1.70.

60 Alexander Zelikovsky. A series of approximation algorithms for the acyclic directed Steiner tree problem. *Algorithmica*, 18(1):99–110, 1997. doi:10.1007/BF02523690.

61 Leonid Zosin and Samir Khuller. On directed Steiner trees. In *Proceedings of ACM-SIAM SODA*, pages 59–63, 2002. URL: http://dl.acm.org/citation.cfm?id=545381.545388.

Directed Low Diameter Decomposition for Structured Digraphs

Shinwoo An
Bar-Ilan University, Ramat Gan, Israel

Arnold Filtser
Bar-Ilan University, Ramat Gan, Israel

Abstract

Low diameter decompositions, or LDDs for short, are a fundamental primitive in the design of efficient graph algorithms. Roughly speaking, an LDD is a distribution over partitions of the vertices into bounded-diameter clusters such that nearby vertices are likely to be clustered together. Recently, there has been growing interest in lifting the notion of LDDs into *directed graphs*. In particular, there are two natural directed analogues. The first is a directed LDD, where after removing a random subset of edges, every strongly connected component has a small diameter. The second is a quasipartition, which imposes the stronger requirement that whenever one vertex can still reach another after the edge removal, the two vertices must be close in the original directed metric. Every quasipartition yields an LDD, but the converse does not necessarily hold.

In this work, we initiate the systematic study of LDDs in structured directed graphs. As our first main result, we show that any directed graph with pathwidth pw admits an $(O(\mathsf{pw}), \Delta)$-LDD. This improves upon the previous best-known $(2^{O(\mathsf{pw}^2)}, \Delta)$-LDD construction, which was implicitly derived from the quasipartition result of Salmasi, Sidiropoulos, and Sridhar [SODA'19].

As our second result, we show that the integrality gap of the Directed Non-Bipartite Sparsest-Cut LP relaxation on an n-vertex graph with treewidth tw is $O(\mathsf{tw} \log n)$. This improves upon the $O(\mathsf{tw} \log^2 n)$ bound of Mémoli, Sidiropoulos, and Sridhar [ICALP'16, Algorithmica'18]. We obtain this result through the refined analysis of the quasipartition construction of Mémoli et al. for bounded treewidth graphs.

2012 ACM Subject Classification Theory of computation → Random projections and metric embeddings

Keywords and phrases Low diameter decomposition, quasipartition, metric embedding, pathwidth

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.16

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2606.31560

Funding The work by An was supported by the Isreal Science Foundation grant #1042/22, Israel Science Foundation grant #800/22, and US-Israel Binational Science Foundation Grant #2022/131. The work by Filtser was supported by the Isreal Science Foundation grant #1042/22.

1 Introduction

Decomposing graphs into well-structured pieces is a central paradigm in the design of graph algorithms. In particular, low diameter decompositions (LDDs), which are a distribution over partitions of the vertices into bounded-diameter clusters such that nearby vertices are likely to be clustered together, have played a central role in undirected graphs. They serve as key primitive underlying a wide range of problems, a partial list of examples include: metric embedding [6, 18, 34, 41, 21], multi-commodity flow [32, 38], Steiner point removal [17], near-linear SDD solvers [10], Lipschitz extension [36], spanners [24, 28], and more. In


© Shinwoo An and Arnold Filtser;
licensed under Creative Commons License CC-BY 4.0

undirected graphs, low diameter decompositions have been extensively studied not only in general graphs, but also in structured graph classes [32, 19, 1, 3, 20, 23, 15], where additional structure often leads to improved guarantees.

Recently, there has been a growing interest in extending low diameter decompositions to *directed graphs* [8, 9, 12, 39], despite the inherent technical challenges compared to the undirected setting. A notable application of this line of work is a recent near-linear time algorithm for the negative-weight single-source shortest path (SSSP) problem [9]. A *strongly connected component* of a directed graph G, an SCC in short, is a maximal subgraph H of G such that for every pair u, v of vertices in H, both u is reachable from v and v is reachable from u. For vertices u, v of G, let $d_G(u, v)$ denote the length of a shortest directed path from u to v in G, with $d_G(u, v) = \infty$ if no such path exists. For a subgraph $H \subseteq G$, the strong diameter of H is $\max_{u,v \in V(H)} d_H(u, v)$, whereas the weak diameter of H is $\max_{u,v \in V(H)} d_G(u, v)$. Here, $V(H)$ denotes the vertex set of H.

▶ **Definition 1** (Directed Low-Diameter Decomposition, [8, 9]). *Given an edge-weighted directed graph $G = (V, E, w)$ and a parameter $\Delta > 0$, a (β, Δ)-LDD is a distribution over subsets $S \subset E$ of edges, called a cutset, such that*

1. *every strongly connected component in $G \setminus S$ has a weak diameter of at most Δ, and*
2. *for every edge $e \in E$, $\Pr[e \in S] \leq \beta \cdot \frac{w(e)}{\Delta}$.*

We call β the loss parameter.

On the other hand, a parallel line of work has focused on *quasipartitions* for directed graphs [40, 31, 43], motivated by their connection to the integrality gap of the directed sparsest cut problem and the directed multicut problem [13].

▶ **Definition 2** ((Random) Quasipartition [40]). *Given an edge-weighted directed graph $G = (V, E, w)$ and a parameter $\Delta > 0$, a (β, Δ)-quasipartition is a distribution over a cutset $S \subseteq E$ such that*

- *for every u-v path in $G \setminus S$, $d_G(u, v) \leq \Delta$, and*
- *for every edge $e \in E$, $\Pr[e \in S] \leq \beta \cdot \frac{w(e)}{\Delta}$.*

We call β the loss parameter.

Notably, quasipartitions impose strictly stronger constraints than LDDs, and thus any (β, Δ)-quasipartition immediately yields a (β, Δ)-LDD, but not vice versa. Intuitively, in the quasipartition construction, one has to enforce that the distance between endpoints of every valid path is at most Δ. In contrast, the LDD construction only restricts the distance between endpoints of the paths within a *strongly connected component*. One specific example is directed acyclic graphs (DAGs), which are directed graphs avoiding directed cycles. It is easy to observe that every DAG itself is a $(0, \Delta)$-LDD for every parameter $\Delta > 0$ since its SCCs are isolated singletons. On the other hand, constructing a quasipartition with a small loss parameter on a DAG remains highly non-trivial, as a DAG still might contain arbitrarily long paths that must be cut. This separation is also reflected in a polynomial gap over the achievable loss parameters. While every directed graph on n vertices admits an $(O(\log n \cdot \log \log n), \Delta)$-LDD [12], there exists an n-vertex directed graph that does not admit any $(o(n^{1/7} \log^{4/7} n), \Delta)$-quasipartition [40].

Quasipartitions and LDDs are not merely theoretical concepts. Rather, they play a crucial role in driving distance-preserving graph sparsification scheme. Here, the goal is to obtain a simplified representation of the given graph while approximately preserving its distances. A number of such structures for directed graphs have been studied: including spanners [7], distance preservers [11], hopsets [33], and more. From the aforementioned lower

bound for the quasipartition, [40] showed that one cannot efficiently embed a directed graph into *quasiultrametric space*, which is equivalent to the shortest path metric in directed trees. In contrast, by changing the viewpoint to LDDs, Assadi, Hoppenworth, and Wein [5] showed that one can (stochastically) embed any directed graph into two DAGs with distortion $O(\log^3 n)$. The distortion bound has been improved to $O(\log n \cdot \log\log n)$ by Filtser [22].

In structured directed graph classes, the distinction between the two decompositions remains poorly explored. First, all known LDD results for structured directed graphs have an indirect nature: Instead of being tailored directly to LDDs, they are either obtained via quasipartition constructions (bounded pathwidth and bounded treewidth [40, 43]) or simply bypassed by applying general graph bounds [12] (planar graphs [31]). In addition, there are no techniques that are specifically tailored to constructing LDDs in this setting.

Second, among the prior works that are indirectly derived from the quasipartition constructions, the resulting bounds fail to yield a clear improvement over the general graph bounds. Specifically, the loss parameter either suffers from a polynomial dependence on treewidth [40] or even worse, an exponential dependency on pathwidth [43]. Consequently, these results surpass the general bounds of [12] only when the parameters are strictly sublogarithmic in the graph size.

To overcome these barriers, a natural starting point is to focus on pathwidth [42]- one of the most standard and well-studied graph parameters which intuitively measures the degree to which a given graph resembles a path. Indeed, based on their structural advantage, bounded pathwidth graphs admit a wide range of efficient algorithms for fundamental graph problems [2, 25, 26, 30, 35]. From the perspective of LDDs, such graphs also enjoy improved guarantees on the loss parameter compared to general graphs in the undirected setting [3]. Hence, it is natural to ask whether the same phenomenon holds in the directed setting.

Motivated by these insights and the gap between the undirected and directed settings, we initiate the study of LDDs in structured directed graphs, especially bounded pathwidth graphs, moving beyond those obtained indirectly via quasipartition approaches. Our main result is that every directed graph of pathwidth pw admits an $(O(\mathsf{pw}), \Delta)$-LDD. This significantly improves the previous best-known result of $(2^{O(\mathsf{pw}^2)}, \Delta)$-LDD [40] in this setting.

▶ **Theorem 3.** *Let G be an edge-weighted directed graph and $\mathsf{pw}(G) = k$. Then, for any $\Delta > 0$, there is a $(O(k), \Delta)$-LDD of G.*

Classical constructions of (undirected/directed) LDDs rely on the ball carving technique [6], where we recursively compute clusters by removing regions defined by distance thresholds around a center. In graphs of bounded pathwidth, prior work for undirected graphs [3] bypasses the general bound of [6] via a path-carving approach: repeatedly choosing a shortest path as a backbone and removing a distance-threshold neighborhood around it, exploiting the path-like structure of the graph. Provided that one can efficiently remove this neighborhood [3, 29], this approach yields an improved loss bound compared to [6].

On the other hand, in general directed graphs, one natural difficulty is that, once we carve a nearby region around a center as a cluster, we cannot guarantee the diameter of the cluster. To overcome this difficulty, all prior works that directly compute LDDs [8, 9, 12, 39] recursively apply the ball carving approach for each carved cluster. It seems difficult to eliminate the $\log n$ dependency on the loss parameter under this approach.

Our main technical contribution is a novel carving procedure with respect to a shortest path, which we call *scissors carving*, that avoids this difficulty. In essence, our approach carefully decomposes a shortest path into shorter subpaths, and applies the ball carving to each segment in a "local" sense. The key insight is that, we can efficiently carve away a path

Table 1 Previous and new results on graph partition in directed graphs.

Graph Family	Partition type	Loss parameter β	Reference
General graphs	Undirected LDD	$\Theta(\log n)$ (UB and LB)	[6]
Genus g		$\Theta(\log g)$	[37]
Pathwidth pw		$\Theta(\log \mathsf{pw})$	[3]
Treewidth tw		$\Theta(\log \mathsf{tw})$	[23]
K_r-minor free		$\Theta(\log r)$	[15]
General digraphs	Directed LDD	$O(\log n \cdot \log\log n)$	[12]
General digraphs	Quasipartition	$\Omega(n^{1/7}\log^{4/7} n)$ (LB)	[14, 40]
Planar	Quasipartition	$O(\log^2 n)$	[31]
Treewidth ≤ 2	Quasipartition	$O(1)$	[43]
Treewidth tw	Quasipartition	$O(\mathsf{tw}\log n)$	[40]
Pathwidth pw	Quasipartition	$2^{O(\mathsf{pw}^2)}$	[43]
Pathwidth pw	Directed LDD	$O(\mathsf{pw})$	Theorem 3

whose length is sufficiently short[1]. In addition, we ensure that any carved cluster from such short paths has a small diameter. We believe that our technique provides a first step toward developing LDD techniques tailored to structured directed graphs, and that the technique itself is of independent interest.

As a secondary contribution, we revisit quasipartitions in directed graphs of bounded treewidth [40], and improve the $O(\mathsf{tw}\log^2 n)$ bound on the integrality gap of the directed non-bipartite sparsest cut [40] by a factor of $O(\log n)$, through a better analysis. Previous and new results about the integrality gap are summarized in Table 2. See Section 5 for the problem definitions and details of prior works on it.

▶ **Theorem 4.** *Let G be an n-vertex directed graph of treewidth tw. The integrality gap (also called the flow-cut gap) of the Directed Non-Bipartite Sparsest-Cut LP relaxation on G is $O(\mathsf{tw}\log n)$.*

2 Preliminaries

Throughout this paper, all missing proofs can be found in the full version. For a positive integer n, we denote by $[n]$ the set of all positive integers at most n.

Graph notations. Let $G = (V, E, w)$ be an edge-weighted directed graph, with n vertices. We often use $V(G)$ and $E(G)$ to denote the vertex set and the edge set in G, respectively. For a set A of vertices in G, we use $G \setminus A$ to denote the graph obtained from G by removing all vertices in A and their adjacent edges. We use $G[A]$ to denote a *induced subgraph* of G with respect to A. That is, $G[A] = G \setminus (V \setminus A)$. Similarly, for a set B of edges in G, we use $G \setminus B$ to denote the graph obtained from G by removing all edges in B. Two vertices u, v in

[1] See Section 3.2 for more details.

Table 2 Previous and new results on the integrality gap of the directed non-bipartite sparsest cut LP. Here, $\tilde{O}(\cdot)$ hides polylog n factor.

Graph Family	Integrality Gap	Ref.	Notes
General digraphs	$O(\sqrt{n})$	[27]	UB
	$\tilde{O}(n^{11/23})$	[4]	UB
	$\tilde{\Omega}(n^{1/7})$	[14]	LB
Planar	$O(\log^3 n)$	[31]	UB
Pathwidth pw	$2^{O(\mathsf{pw}^2)} \cdot \log n$	[40]+[43]	UB
Treewidth tw	$O(\mathsf{tw} \log^2 n)$	[40]	UB
Treewidth tw	$O(\mathsf{tw} \log n)$	Theorem 4	UB

G are *strongly connected* if there exist both a u-v path and a v-u path in G. A graph G is *strongly connected* if there is a u-v path for every $u, v \in V$. A *strongly connected component* of G, an SCC in short, is a maximal strongly connected subgraph of G. For a path P in G, the *length* of P is defined as the total edge weight in P. We use $d_G(u, v)$ to denote the *distance* between u and v in G. The *diameter* of G is defined as the maximum value of $d_G(u, v)$ among every pair $(u, v) \in V \times V$.

Treewidth and Pathwidth. A *tree-decomposition* (resp. *path-decomposition*) of an *undirected* graph G is a pair $(T, \{B_t\}_{t \in V(T)})$ consisting of a tree (resp. path) T and a family of sets $\{B_t\}_{t \in V(T)}$ of vertices in G such that

1. $V(G) = \bigcup_{t \in V(T)} B_t$,
2. for every edge $uv \in E(G)$, there exists a node $t \in V(T)$ such that $\{u, v\} \subseteq B_t$, and
3. for every vertex $v \in V(G)$, the set $\{t \in V(T) : v \in B_t\}$ induces a connected subtree (resp. subpath) of T.

The *width* of a tree-decomposition (resp. path-decomposition) $(T, \{B_t\}_{t \in V(T)})$ is defined as $\max_{t \in V(T)} |B_t| - 1$. The *treewidth* $\mathsf{tw}(G)$ (resp. *pathwidth* $\mathsf{pw}(G)$) of a graph G is the minimum width among all tree decompositions (resp. path decompositions) of G.

For a *directed* graph $G = (V, E)$, we consider the undirected graph $\bar{G}$ obtained by ignoring the edge directions and removing any parallel edges. Formally speaking, $\bar{G}$ has a vertex set $V(\bar{G}) = V$ and an edge set $E(\bar{G}) = \{\{u, v\} \mid (u, v) \in E \text{ or } (v, u) \in E\}$. We say that G has treewidth (resp. pathwidth) k if its underlying undirected graph $\bar{G}$ has treewidth (resp. pathwidth) k.

3 Technical Overview

Low diameter decomposition (LDD) for undirected graphs. Let G be an undirected graph. A (β, Δ)-LDD of G is a distribution over vertex partitions such that in every partition, each cluster has a diameter of at most Δ, and for every vertex pair u, v, the probability that u, v belong to different clusters is at most $\beta \cdot \frac{d_G(u,v)}{\Delta}$. We call β the *loss parameter*. Every n-vertex graph admits an $(O(\log n), \Delta)$-LDD for any $\Delta > 0$, which is tight [6].

Ball carving is a classical technique for computing an LDD. For a vertex $u \in V$ and parameter $R > 0$, a *ball* $B_G(u, R)$ is defined as the set of vertices $v \in V$ such that $d_G(u, v) \leq R$. We call R the *radius* of $B_G(u, R)$. The algorithm of Bartal [6] repeatedly takes a vertex w from the remaining graph, samples a radius R from an exponential distribution with the

parameter $\Theta(\log n/\Delta)$, and removes the ball $B_G(w, R)$ from the graph as a single cluster. Using the memoryless property of the exponential distribution, Bartal [6] showed that the algorithm yields an $(O(\log n), \Delta)$-LDD.

Low diameter decomposition for bounded pathwidth graphs. Abraham et al. [3] showed that if G has pathwidth pw, it admits an $(O(\log \mathsf{pw}), \Delta)$-LDD. While this is a special case of a more general framework for minor-free graphs, we sketch their approach for the bounded pathwidth graphs. The following lemma was implicitly proved by several works (e.g., [3, 29]).

▶ **Lemma 5.** *Let G be an undirected graph and P be a shortest path in G. For any $\Delta > 0$, there is a set $N \subset P$ of vertices, called a net, such that*
1. *(Covering) $P \subset \bigcup_{v \in N} B_G(v, \Delta)$, and*
2. *(Sparsity) for every $u \in G$, $|B_G(u, 2\Delta) \cap N| = O(1)$.*

Their algorithm works in several rounds. In each round, for each connected component in the current graph, we pick a shortest path P connecting a vertex in the first bag and a vertex in the last bag (in a path decomposition), and compute a net N of that path using Lemma 5. Next, for each $v \in N$, we sample a radius R_v from a *truncated exponential distribution* with a parameter λ over the range $[\frac{\Delta}{4}, \frac{\Delta}{2}]$ so that each cluster has a diameter at most Δ. That is, the exponential distribution with the parameter λ conditioned on the event that the output belongs to $[\frac{\Delta}{4}, \frac{\Delta}{2}]$. We will define λ later. Finally, we carve $B_G(v, R_v)$ from the current graph as a single cluster.

Crucially, due to the covering property of Lemma 5, all vertices of P are carved away after the carving step. Moreover, since P is a path connecting a vertex in the first bag and a vertex in the last bag, P intersects every bag of the path decomposition. Therefore, the pathwidth of the graph decreases by at least one. Thus, the algorithm terminates after pw rounds. Due to the sparsity property of Lemma 5, we can show that for every pair $u, v \in V$, at most $\nu = O(\mathsf{pw})$ balls, called threateners, affect the probability that u, v belong to different clusters.

Roughly speaking, the truncated exponential distribution behaves similarly to the exponential distribution, i.e., retaining a memoryless property, up to a small error. These errors from the sampled radii of different threateners are accumulated. In order to keep the total error negligible, the parameter should be at least logarithmic in the number of threateners. In light of this, Abraham et al. [3] set the parameter $\lambda = \Theta(\frac{\log \nu}{\Delta}) = \Theta(\frac{\log \mathsf{pw}}{\Delta})$, yielding an $(O(\log \mathsf{pw}), \Delta)$-LDD of G. This path-carving framework is the key ingredient that we will adapt to the directed setting later.

Low diameter decomposition for general directed graphs. A directed analogue of the low diameter decomposition (Definition 1) was very recently introduced by Bernstein, Gutenberg, and Wulff-Nilsen [8] (though not stated explicitly), who showed that every n-vertex directed graph G admits an $(O(\log^2 n), \Delta)$-LDD for every parameter $\Delta > 0$. Later, Bernstein, Nanongkai, and Wulff-Nilsen [9] explicitly defined the notion of a low diameter decomposition and presented a simpler algorithm yielding an $(O(\log^2 n), \Delta)$-LDD. This bound has been improved to $O(\log n \cdot \log \log n)$ by Bringmann et al. [12].

The algorithm of Bernstein et al. [9] follows the ball carving framework: it samples a radius and carves either an in-ball or an out-ball[2] around a chosen vertex. A key difficulty in the directed setting is that, unlike the undirected case, one cannot guarantee the weak

[2] We defer the definitions of an in-ball and an out-ball to Section 3.2

diameter bound of an SCC contained in a single ball. To resolve this, the algorithm of [9] recursively applies the same carving procedure to every SCC until all SCCs have a small weak diameter. By carefully selecting centers, Bernstein et al. [9] managed to bound the recursion depth by $O(\log n)$. This yields an $O(\log^2 n)$ bound on the loss parameter.

3.1 Low diameter decomposition for special classes of directed graphs

The aforementioned issue remains even for special classes of directed graphs, such as those with bounded pathwidth, bounded treewidth, or planar graphs. We briefly review some prior works that circumvented this issue. We remark that while all prior work primarily addressed random quasipartitions for those graph classes, their quasipartition results directly translate to low diameter decompositions with the same loss parameter[3]. Consequently, we present the prior work from the viewpoint of low diameter decompositions.

Separator-based approach. Mémoli, Sidiropoulos, and Sridhar [40] showed that every n-vertex directed graph with treewidth tw admits an $(O(\mathsf{tw}\log n), \Delta)$-LDD for every parameter $\Delta > 0$. Their approach is based on a balanced separator. We call $U \subset V(G)$ a *balanced separator* if the number of vertices in each connected component of $G \setminus U$ is at most $\frac{2}{3}|V(G)|$. It is well known that every graph G with treewidth tw admits a *balanced separator* $U \subset V(G)$ of size at most $\mathsf{tw}+1$.

The algorithm of [40] computes a balanced separator U, samples a radius, and constructs both in-balls and out-balls around each vertex of U. Next, it adds the boundary edges of these balls to the cutset S. Then, it recursively proceeds on each SCC of $G \setminus U$. Critically, any SCC of $G \setminus S$ is either disjoint from U or has a weak diameter of at most Δ. The loss parameter at a single recursion level is $O(\mathsf{tw})$ since the algorithm computes a cutset through $2|U| = O(\mathsf{tw})$ balls. In addition, the recursion depth is $O(\log n)$. Overall, the algorithm of [40] yields an $(O(\mathsf{tw}\log n), \Delta)$-LDD.

Kawarabayashi and Sidiropoulos [31] showed that every n-vertex planar directed graph admits an $(O(\log^2 n), \Delta)$-LDD. Instead of a (balanced) vertex separator, they utilized the shortest path separator of [44], which consists of a constant number of shortest paths. Again, the $O(\log n)$ factor arises from the fact that the recursion depth is $O(\log n)$. We remark that it seems difficult to avoid the $O(\log n)$ factor in any separator-based approach since the recursion depth is $O(\log n)$ and the edge-cut probability incurred in each round must be handled independently.

LDD of constant loss parameter for bounded pathwidth directed graphs. Salmasi, Sidiropoulos, and Sridhar [43] showed that every n-vertex directed graph G with pathwidth pw admits a $(2^{O(\mathsf{pw}^2)}, \Delta)$-LDD for every parameter $\Delta > 0$. This is the only known LDD result for structured directed graphs where the loss parameter is independent of n.

We briefly sketch their approach at a high level. We start by setting the current graph to $H = G$. In each round, [43] computes a u-v shortest path P that connects the first and last bags of the path decomposition. Instead of carving the balls around the net vertices along P, the algorithm computes a random shift $z \in [0, \Delta]$ and adds all edges crossing distances of $i\Delta + z$ from u for every integer i, to the cutset. Notably, after this, they remove the edge set of P from H (recall that in the ball carving framework, the cutset is removed) and recurse. Finally, the algorithm repeats the entire procedure in the reversed direction. The recursion depth is $O(\mathsf{pw}^2)$.

[3] From Definitions 1 and 2, it is easy to check that every (β, Δ)-quasipartition is a (β, Δ)-LDD.

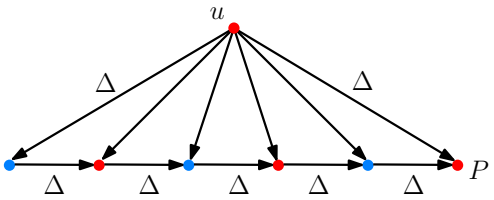


Figure 1 Counter-example of Lemma 5 for directed graphs. Consider a directed graph G consisting of a shortest path P of n edges of weight Δ, an apex vertex u, and an edge (u, p) of weight Δ for every vertex $p \in P$. Then, for any subset $N \subseteq P$ that satisfies the covering property, i.e., $P \subset \bigcup_{v \in N} B_G^+(v, \Delta)$, notice that each ball covers at most $O(1)$ vertices of P. Therefore, $|N| = \Theta(n)$. The blue points indicate the points in N. Since $d_G(u, v) = \Delta$ for every $v \in P$, $N \subseteq B_G^+(u, 2\Delta)$.

Salmasi, Sidiropoulos, and Sridhar [43] managed to eliminate the dependency on n by carving away the boundary edges of concentric layers with a random shift. On the other hand, they remove the edge set of P instead of the cutset edges. This causes a more involved analysis of bounding the weak diameter, and yields a $(2^{O(\mathsf{pw}^2)}, \Delta)$-LDD. We also remark that their analysis crucially utilizes the structure of the bounded pathwidth graphs, making it seemingly difficult to lift their result to bounded treewidth graphs or planar graphs.

3.2 Our solution: Scissors Carving

Our algorithm departs from the previous approaches. We develop a new carving procedure, which we call *scissors carving*. This is our main technical contribution that enables us to achieve an $(O(\mathsf{pw}), \Delta)$-LDD. Let $G = (V, E, w)$ be an edge-weighted directed graph. For a vertex $u \in V$ and a parameter $R > 0$, we define the *out-ball* $B_G^+(u, R)$ and the *in-ball* $B_G^-(u, R)$ as

$$B_G^+(u, R) := \{x \in V \mid d_G(u, x) \leq R\}, \text{ and } B_G^-(u, R) := \{x \in V \mid d_G(x, u) \leq R\},$$

respectively. We call R the *radius*.

At a high level, suppose we are given a target x-y path P that we want to carve away. We compute an out-ball around x and an in-ball around y with radii proportional to $d_G(x, y)$, and then remove their boundary edges. Then, we recurse on each SCC intersecting P.

The key insight is that this approach always makes progress for each recursive SCC: either the length of the portion of P contained in each SCC, called the *forward distance*, is reduced, or the pathwidth of the SCC is decreased. Once the length of the current path becomes sufficiently small, we apply the same procedure in the reverse direction, i.e., computing an in-ball around x and an out-ball around y, to reduce the *backward distance* $d_G(y, x)$. The separation between the forward-reduction stage and the backward-reduction stage is essential, as reducing the backward distance using two balls is only possible once the forward distance is sufficiently small.

We demonstrate the two advantages of the scissors carving framework as follows. First, we recursively compute two balls of geometrically decreasing radii. Even though we may compute many balls across the recursion, most of them have radii much larger than Δ. This allows us to bound the edge cut probability by $\Pr[e \in S] \leq O(1) \cdot \frac{w(e)}{\Delta}$.

Second, after both the forward-reduction stage and the backward-reduction stage, we ensure that both the forward distance and the backward distance of each remaining SCC H are sufficiently small. Using this structural advantage, we show that any SCC of H fully contained in a single ball of radius Δ around $u \in H$ has a weak diameter of at most $O(\Delta)$.

This overcomes the limitation that, in general directed graphs, an SCC contained in a single ball of radius Δ can have an arbitrarily large diameter. This allows us to ensure the desired diameter bound.

Based on the scissors carving procedure, we compute an $(O(\mathsf{pw}), \Delta)$-LDD following the framework of [3]. Specifically, we start by setting the current graph to $H = G$. At each stage, we take a shortest path P in H that we want to carve away. In particular, we choose P such that $\mathsf{pw}(H \setminus P) < \mathsf{pw}(H)$. Then, we apply scissors carving to P and update the cutset S. This ensures that every SCC of $H \setminus S$ either has a bounded weak diameter or a reduced pathwidth (See Lemma 6). Furthermore, we shall guarantee that every SCC disjoint from P has a reduced pathwidth. Then, we recurse on every SCC of $H \setminus S$ whose weak diameter is greater than Δ. By taking the union S of the cutsets computed from all recursive calls, we ensure that every SCC of $G \setminus S$ has a weak diameter of at most Δ. The recursion depth is at most $\mathsf{pw}(G)$. Due to the union bound of probability, the probability bound of Lemma 6 yields $\Pr[e \in S] \leq O(\mathsf{pw}(G)) \cdot \frac{w(e)}{\Delta}$.

▶ **Lemma 6** (informal).[4] *Let H be a strongly connected subgraph of G. There is a randomized algorithm that computes a cutset $S \subset E(H)$ such that for every strongly connected component C of $H \setminus S$, either $\mathsf{pw}(C) < \mathsf{pw}(H)$ or the weak diameter of C is $O(\Delta)$. Moreover, for every $e \in E(H)$, $\Pr[e \in S] \leq O(1) \cdot \frac{w(e)}{\Delta}$.*

In the rest of this section, we explain the core idea of the scissors carving technique and give a proof sketch of Lemma 13, especially for the first statement. We begin by defining some notation used in this paper. For a vertex set $C \subseteq V$, we denote the set of edges whose source endpoint is in C and target endpoint is outside C by $\delta^+(C)$. Similarly, we denote the set of edges whose source endpoint is outside C and target endpoint is inside C by $\delta^-(C)$.

Let H be a subgraph of G and P be a shortest path in G whose two endpoints are contained in H. For a strongly connected *subgraph* C of H, let u and v be the first and the last vertices of $P \cap C$ (along the order of P), respectively. We define the *surviving subpath* of P with respect to C, denoted as $P[C]$, as the u-v subpath of P. Note that $P[C]$ may not be fully contained in C. We define the *forward distance* and the *backward distance* of C (with respect to P) as $d_G(u, v)$ and $d_G(v, u)$, respectively, and denote them by $f_P(C)$ and $b_P(C)$, respectively. Note that we analyze these distances with respect to the shortest paths in G, rather than the current subgraph C. This is sufficient for our purpose since our goal is to bound the *weak diameter* of each SCC.

We use the following lemma. See Figure 2 for an illustration.

▶ **Lemma 7.** *Let $B_1 = B^*_G(v, R)$ and $B_2 = B^{*'}_G(v, R)$ with $*, *' \in \{+, -\}$, and let C be a strongly connected subgraph of G. Then,*

- *every SCC of $C \setminus (\delta^*(B_1) \cup \delta^{*'}(B_2))$ is either fully contained in $B_1 \cap B_2$ or disjoint from $B_1 \cap B_2$, and*
- *every SCC of $C \setminus (\delta^*(B_1) \cup \delta^{*'}(B_2))$ is either fully contained in B_1, or fully contained in $B_2 \setminus B_1$, or disjoint from $B_1 \cup B_2$.*

Next, we explain the scissors carving procedure. We are given a strongly connected subgraph H of G and a shortest path P as input, such that the two endpoints of P are contained in H, and $\mathsf{pw}(H \setminus P) < \mathsf{pw}(H)$. The algorithm consists of three stages, and the first two stages operate recursively. As we will see below, these stages aim to reduce the forward distance, reduce the backward distance, and bound the weak diameter of each SCC, respectively.

[4] Here, we give an informal statement in order to hide several technical terms behind it. See Lemma 13 for details.

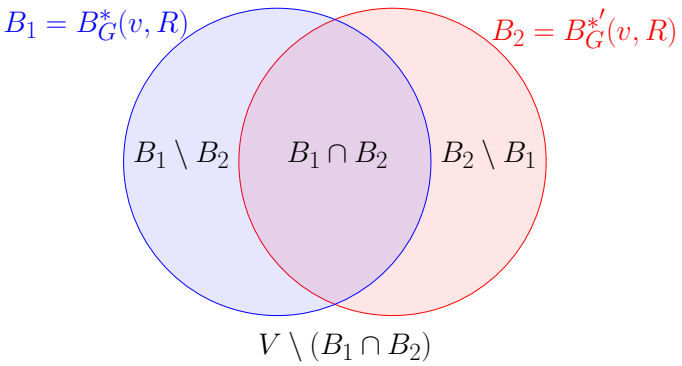


Figure 2 Due to the definition of $\delta^*(\cdot)$, every SCC of $H \setminus \delta^*(B_1)$ is either contained in B_1 or disjoint from B_1. The same holds symmetrically for B_2. Therefore, every SCC of $H \setminus (\delta^*(B_1) \cup \delta^{*'}(B_2))$ is fully contained in one of the regions of the Venn diagram. This proves both items of Lemma 7 for the special case of $C = G$. See Section 4 for the general case.

First stage: reducing forward distance. In the first stage we are given a strongly connected subgraph C of H as input. Let $P[C] = \langle u, \ldots, v \rangle$ and $\tau = f_P(C)$. We sample the radius R uniformly at random from $[\frac{1}{2}\tau, \frac{2}{3}\tau]$, and compute two balls $B_1 = B^+_G(u, R)$, $B_2 = B^-_G(v, R)$ and add their boundary edges $\delta^+(B_1) \cup \delta^-(B_2)$ into the cutset S. We then recursively apply this approach to each SCC of $C \setminus S$ contained in $B_1 \cup B_2$. We halt the first stage when $f_P(C) \leq \Delta$, and proceed to the second stage. This completes the first stage.

We show that the above procedure ensures that each resulting SCC for the recursive calls has a reduced forward distance, while the remaining SCCs have a reduced pathwidth. Hence, it suffices to proceed with the SCCs of the former type, i.e., those with a reduced forward distance. To see this, since we take R from the range $[\frac{1}{2}\tau, \frac{2}{3}\tau]$ and $P[C]$ is a shortest path in G, for every $w \in P[C]$, either $d_G(u, w) \leq R$ or $d_G(w, v) \leq R$. Hence, $P \subseteq B_1 \cup B_2$. In addition, due to Lemma 7, every SCC of $C \setminus S$ is either fully contained in B_1, or fully contained in $B_2 \setminus B_1$, or disjoint from $B_1 \cup B_2$. Let C' be an SCC of $C \setminus S$ and $P[C'] = \langle x, \ldots, y \rangle$. Since $P[C']$ is a subpath of P and $C' \subseteq C$, $P[C']$ is a subpath of $P[C]$. If C' is contained in B_1,

$$d_G(x, y) \leq d_G(u, y) \overset{B_1}{\leq} R \leq \frac{2}{3}\tau.$$

Otherwise, if C' is contained in $B_2 \setminus B_1$, we have that

$$d_G(x, y) \leq d_G(x, v) \overset{B_2}{\leq} R \leq \frac{2}{3}\tau.$$

Finally, if C' is disjoint from $B_1 \cup B_2$, $P \subseteq B_1 \cup B_2$ implies that C' is a subset of $C \setminus P$. Then, we have that $\mathsf{pw}(C') \leq \mathsf{pw}(C \setminus P) \leq \mathsf{pw}(H \setminus P) < \mathsf{pw}(H)$. Hence, we conclude that every SCC of $G \setminus S$ either has a reduced forward distance or a reduced pathwidth, as desired.

Second stage: reducing backward distance. In the second stage, we are given a strongly connected subgraph C of H as input with the additional property that $f_P(C) \leq \Delta$. Let $P[C] = \langle u, \ldots, v \rangle$ and $b_P(C) = \tau'$. Importantly, throughout this stage, we ensure that the condition on $f_P(C)$ always holds. Our algorithm is almost identical to that of the first stage, except that we compute two balls in the reversed direction and sample the radius R from a narrower interval. The reason we choose a narrower interval is to reduce the backward distance by a constant fraction.

More precisely, we sample R uniformly at random from $[\frac{1}{2}\tau', \frac{13}{24}\tau']$, compute two balls $B_1 = B_G^-(u, R), B_2 = B_G^+(v, R)$ and add $\delta^-(B_1) \cup \delta^+(B_2)$ to the cutset S. We recursively apply this approach to each SCC of $C \setminus S$ contained in $B_1 \cup B_2$. We halt the second stage when $b_P(C) \leq 8\Delta$. Here, 8 is also a carefully chosen constant in order to bound the backward distance.

We show that a symmetrical property holds for this stage: Every SCC contained in $B_1 \cup B_2$ has a reduced backward distance, and every SCC disjoint from $B_1 \cup B_2$ has a reduced pathwidth. Let C' be an SCC of $C \setminus S$ and $P[C'] = \langle x, \ldots, y \rangle$. First, we bound $b_P(C')$ in the case where $C' \subseteq B_1 \cup B_2$. From Lemma 7, without loss of generality, let us assume $C' \subseteq B_1$. The other case where $C' \subseteq B_2 \setminus B_1$ is analogous. From the argument of the first stage, $P[C']$ is a subpath of $P[C]$ and $d_G(u, v) = f_P(C) \leq \Delta$. In addition, due to the triangle inequality, we have that:

$$d_G(y, x) \overset{\triangle}{\leq} d_G(y, u) + d_G(u, x) \leq d_G(y, u) + d_G(u, v) \overset{B_1}{\leq} R + d_G(u, v) \leq R + \Delta \leq \frac{13}{24}\tau' + \Delta.$$

Since the halting condition is $\tau' \leq 8\Delta$, we have that $\frac{13}{24}\tau' + \Delta \leq \frac{2}{3}\tau'$ throughout the second stage. Hence, the backward distance is reduced by $2/3$.

Next, we bound the pathwidth of C' in the case where C' is disjoint from $B_1 \cup B_2$. The difference is that $P[C]$ may not be fully contained in $B_1 \cup B_2$. For simplicity, let us assume that u and v are the leftmost and rightmost vertices of P in the path decomposition, respectively. That is, for two bags X_u and X_v containing u and v respectively, every bag X that intersects P, lies between X_u and X_v in the path decomposition[5]. We consider a shortest v-u path P' of length τ' in G. Since $R \geq \frac{1}{2}\tau'$, it is easy to check that $P' \subseteq B_1 \cup B_2$. Moreover, since P' intersects every bag X between X_u and X_v, P' intersects every bag X that intersects P. Since C' is disjoint from $B_1 \cup B_2$, it is disjoint from P' as well. Hence, for each bag X intersecting $P \cap C'$, we have that:

$$|C' \cap X| < |C' \cap X| + |P' \cap X| = |(C' \cup P') \cap X| \leq |C \cap X|.$$

This establishes that $\mathsf{pw}(C') < \mathsf{pw}(C)$, as desired. See Figure 3(b) for an illustration.

Third stage: reducing the weak diameter. Finally, the third stage serves as the base case. We are given a strongly connected subgraph C of H as input, where the two previous stages guarantee that $f_P(C) \leq \Delta$ and $b_P(C) \leq 8\Delta$. One might ask whether $f_P(C), b_P(C) = O(\Delta)$ directly implies that the weak diameter of C is $O(\Delta)$. This is not true in general. What we can guarantee is only the (bi-)distance between two endpoints of $P[C]$, which does not necessarily bound the distance between every pair of vertices in C.

Let $P[C] = \langle u, \ldots, v \rangle$. We compute two balls $B_G^+(u, R_1)$ and $B_G^-(u, R_2)$ around the same vertex u, where we sample R_1, R_2 uniformly at random from $[\Delta, 2\Delta]$ and $[9\Delta, 10\Delta]$, respectively. Then, we add $\delta^+(B_1) \cup \delta^-(B_2)$ into the cutset S. Every SCC contained in $B_1 \cap B_2$ has a small weak diameter, since for every pair of vertices in $B_1 \cap B_2$, we can draw a detour of length $O(\Delta)$ through u. In addition, it is straightforward to verify that $P[C] \subset B_1 \cap B_2$. Subsequently, every SCC disjoint from $B_1 \cap B_2$ has a reduced pathwidth. Due to Lemma 7 we conclude that every SCC of $G \setminus S$ after the third stage either has a weak diameter of at most Δ or a reduced pathwidth. This completes the proof sketch of Lemma 6.

[5] Indeed, we can guarantee this assumption by carefully taking certain subpaths in each recursion step. See Section 4 for details.

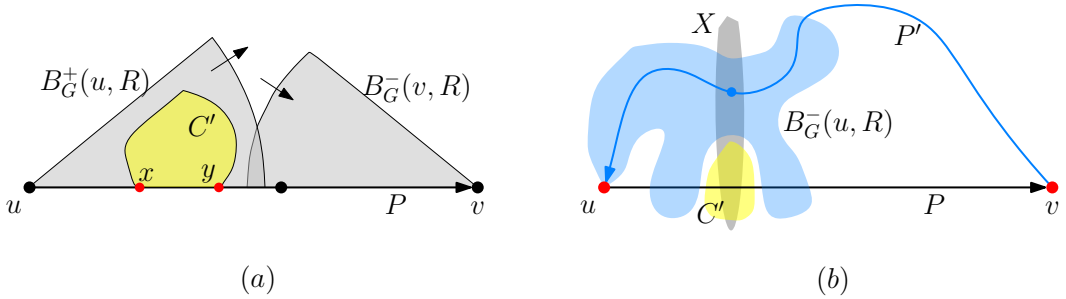


Figure 3 (a) Illustration of the first stage. The gray region denotes two balls $B_G^+(u,R)$ and $B_G^-(v,R)$ where R is sampled from $[\frac{1}{2}\tau, \frac{2}{3}\tau]$ with $\tau = d_G(u,v)$. For an SCC C' (yellow region) contained in $B_1 = B_G^+(u,R)$ and $x, y \in P \cap C'$, we have that $d_G(x,y) \leq d_G(u,y) \leq \frac{2}{3}\tau$, since P is a shortest path in G. (b) Illustration of the second stage. The blue region denotes $B_G^-(u,R)$, and P' represents the shortest v-u path in G. In addition, the yellow region denotes an SCC C' which is disjoint from $B_G^-(u,R)$. Let us assume that u and v are the leftmost and rightmost vertices of P with respect to the path decomposition. Then, for every bag X that intersects $P \cap C'$, X also intersects P' (blue vertex). Since C' is disjoint from P', we can derive $\mathsf{pw}(C') < \mathsf{pw}(C)$.

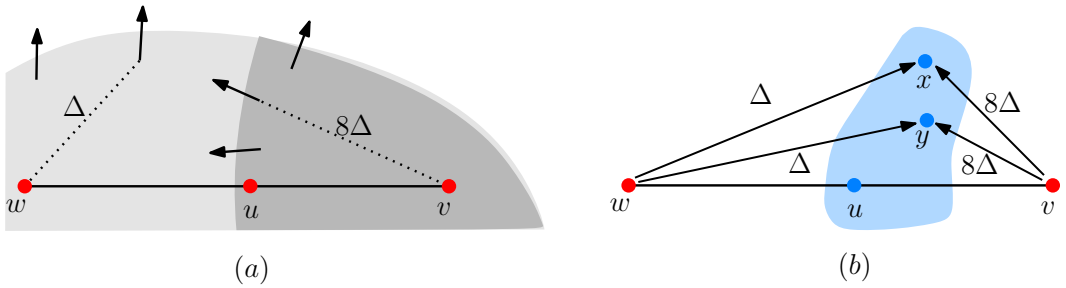


Figure 4 (a) Illustration of the graph partition from the first two stages. In the first stage, an out-ball $B_G^+(w,\Delta)$ (light gray region) containing u, v is computed. In the second stage, an out-ball $B_G^+(v,8\Delta)$ (dark gray region) containing u is computed. (b) At the start of the third stage, an SCC (blue region) might have a large diameter, as no upper bounds for $d_G(x,u)$ and $d_G(y,u)$ are assumed. In the third stage, we reduce the weak diameter of an SCC intersecting P by computing two balls of radii $O(\Delta)$ around u.

▶ Remark. One may expect a logarithmic loss parameter of $O(\log(\mathsf{ppw}(G)))$ as in [3]. However, our scissors carving framework differs from previous ball carving in the following sense. In previous work, once the interior of a ball was carved away, the resulting cluster became "safe", meaning that it was no longer affected by further ball carving steps. This allowed [3] to sample the radius from a (truncated) exponential distribution and utilize its (almost) memoryless property, ensuring them to bound the loss parameter by the logarithmic number of threateners.

In contrast, in our scissors carving, once we compute two balls that cover a shortest path, the interiors of these balls are conveyed to the next round. That is, each edge can be threatened multiple times regardless of whether it is inside a ball or not. Thus, the cut probability becomes linear in the number of threateners. We also remark that the same phenomenon is observed in the directed treewidth case [40].

4 Directed LDDs for Bounded-Pathwidth Directed Graphs

▶ **Theorem 3.** *Let G be an edge-weighted directed graph and $\mathsf{pw}(G) = k$. Then, for any $\Delta > 0$, there is a $(O(k), \Delta)$-LDD of G.*

In this section, we prove Theorem 3. While we state our result in terms of pathwidth, we find it more convenient to use the notion of *path partition-width* (see Definition 8 below). This is a directed analogue of the (undirected) tree partition-width [16] for the directed graphs with small pathwidth.

▶ **Definition 8** (Path Partition)**.** *A* path partition *of a directed graph $G = (V, E)$ is a path $\mathcal{P}$ whose vertices are bijectively associated with the sets of a partition $\mathcal{X} = \{X_1, X_2, \ldots, X_m\}$ of V, called* bags*, such that for every $(u, v) \in E$, there exist $X_i, X_{i+1} \in \mathcal{X}$ for some $1 \leq i \leq m-1$ such that $\{u, v\} \subset X_i \cup X_{i+1}$. The* width *of $\mathcal{X}$ is $\max_{i \in [m]}\{|X_i|\}$. The path partition-width $\mathsf{ppw}(G)$ of G is the minimum width among all path partitions of G.*

To begin with, we apply an edge-preserving isometric embedding from a directed graph with bounded pathwidth into another directed graph with bounded path partition-width. Note that our embedding is not new. Filtser et al. [23] defined a similar embedding from (undirected) bounded treewidth graphs into graphs of bounded tree partition-width[6] graphs. Although not stated explicitly, their embedding is also an edge-preserving isometric embedding.

▶ **Definition 9.** (Edge-preserving isometric embedding)[23] *Let $G = (V, E, w)$, $H = (U, E_H, w_H)$ be directed graphs. We say that a map $\phi : V \to U$ is an* edge preserving isometric embedding *if the following conditions hold:*

- *(isometry) $d_G(u, v) = d_H(\phi(u), \phi(v))$ for every $u, v \in V$,*
- *(edge preservation) there is an injective mapping $\bar{\phi} : E \to E_H$ such that for every $e \in E$, $w(e) = w_H(\bar{\phi}(e))$ and $\bar{\phi}(e)$ belongs to a shortest $\phi(u)$-$\phi(v)$ path in H.*

While the ultimate goal is to design an LDD for directed graphs of bounded pathwidth, our construction algorithm is formally defined on graphs of bounded path partition-width. We remark that a similar approach was used by Filtser et al. [23] to compute an LDD for (undirected) graphs of bounded treewidth. There is an edge-preserving isometric embedding from graphs with bounded pathwidth into graphs with bounded path partition-width. In addition, an LDD of the embedded graph can be transferred back to the original graph without any additional loss.

▶ **Lemma 10.** *Let $G = (V, E, w)$ be a directed graph. Then, there is an edge-preserving isometric embedding from G into a directed graph $H = (U, E_H, w_H)$ such that $\mathsf{ppw}(H) \leq \mathsf{pw}(G) + 1$.*

▶ **Lemma 11.** *Let ϕ be an edge-preserving isometric embedding from G into H, and $\mathcal{D}$ be a (β, Δ)-LDD of H. Then there is a (β, Δ)-LDD of G.*

Based on Lemmas 10 and 11, it suffices to show the existence of $(O(k), \Delta)$-LDD for the directed graphs with path partition-width at most k. Throughout this section, we let G be a directed graph, and let $(\mathcal{P}, \mathcal{X} = \{X_1, X_2, \ldots, X_m\})$ be the path partition of G with width $\mathsf{ppw}(G)$. If it is clear from the context, we slightly abuse the notation and simply

[6] Tree partition can be viewed as a natural generalization of path partition, in the same way that tree decomposition generalizes path decomposition.

denote the path partition $(\mathcal{P}, \mathcal{X})$ by $\mathcal{P}$. Furthermore, we may assume that G is strongly connected as otherwise, we can independently execute our algorithm on each SCC of G. For a subgraph H of G, we define the *induced path partition* of H with respect to $\mathcal{P}$ as $(\mathcal{P}, \{X_i \cap V(H) \mid X_i \in \mathcal{X}\})$, and denote it by $\mathcal{P}[H]$. We denote the width of $\mathcal{P}[H]$ by $w(\mathcal{P}[H])$. Note that for every subgraph H of G, $\mathsf{w}(\mathcal{P}[H]) \leq \mathsf{w}(\mathcal{P}[G]) = \mathsf{ppw}(G)$ trivially holds.

The remainder of this section is devoted to proving the following lemma.

▶ **Lemma 12.** *For any $\Delta > 0$, there is an $(O(\mathsf{ppw}(G)), \Delta)$-LDD of G.*

Next, we present a randomized recursive algorithm Directed-LDD, which receives as input a directed graph $G = (V, E, w)$, a path partition $\mathcal{P}$ of G, a strongly connected subgraph H of G, and a parameter $\Delta > 0$. In addition, Directed-LDD computes a (randomized) cutset of H whose distribution induces a $(O(\mathsf{w}(\mathcal{P}[H])), \Delta)$-LDD of H. See Algorithm 1 for the pseudocode. The algorithm immediately terminates if $\mathsf{w}(\mathcal{P}[H]) = 0$, which implies that H is an empty subgraph. For the other case where $\mathsf{w}(\mathcal{P}[H]) > 0$, Algorithm 1 calls Scissors-Carving algorithm[7], whose function is summarized in the following lemma. We defer the detailed description of Scissors-Carving to Sections 4.1.

▶ **Lemma 13.** *There is an algorithm Scissors-Carving$(G, \mathcal{P}, H, \bar{P}, \Delta)$ which computes a (random) cutset S of $E(H)$ such that:*

1. *For every strongly connected component C of $H \setminus S$, either $\mathsf{w}(\mathcal{P}[C]) < \mathsf{w}(\mathcal{P}[H])$ or the weak diameter of C is at most 12Δ, and*
2. *for every edge e in $E(H)$, $\Pr[e \in S] \leq O(1) \cdot \frac{w(e)}{\Delta}$.*

Then, for each strongly connected component C of $H \setminus S$ such that the weak diameter of C is greater than Δ, the algorithm executes a recursive call Directed-LDD $(G, \mathcal{P}, C, \Delta)$ to compute a random cutset S_C of C. Finally, Directed-LDD returns the union of S and all cutsets S_C returned by the recursive calls. This completes the description of Directed-LDD.

Algorithm 1 Directed-LDD$(G, \mathcal{P}, H, \Delta)$.

```
if w(P[H]) = 0 then
    return ∅
S ← Scissors-Carving(G, P, H, P̄, Δ)
C ← Collection of SCCs of H \ S whose weak diameter is greater than 12Δ
for all C ∈ C do
    S_C ← Directed-LDD(G, P, C, Δ)
    S ← S ∪ S_C
return S
```

Let $S = \text{Directed-LDD}(G, \mathcal{P}, H, \Delta)$.

▶ **Lemma 14.** *The weak diameter of each SCC of $H \setminus S$ is at most 12Δ.*

▶ **Lemma 15.** *For every edge e in H, $\Pr[e \in S] \leq O(\mathsf{w}(\mathcal{P}[H])) \cdot \frac{w(e)}{\Delta}$.*

Finally, Lemma 12 follows from applying Lemmas 14 and 15 by taking $H \leftarrow G$ and $\Delta \leftarrow \frac{\Delta}{12}$.

[7] The algorithm receives $\bar{P}$ as an additional input parameter, where $\bar{P}$ is a sequence of vertices in G satisfying certain conditions. We provide the details of $\bar{P}$ in Section 4.1.

4.1 Reducing path partition-width

In this section, we explain our key subroutine Scissors-Carving. To this end, we define some additional notation that will be used in this section. Recall that $\mathcal{P} = (\mathcal{P}, \mathcal{X})$ consists of m bags $X_1, X_2, \dots X_m$. Let H be a strongly connected subgraph of G. We say an index $i \in [m]$ is a *critical* index of H if $|V(H) \cap X_i| = \mathsf{w}(\mathcal{P}[H])$. Equivalently, we refer to X_i as a *critical bag* of H. Let $\mathcal{C}(H) = \{j_1, j_2, \dots, j_\ell\}$ be the set of critical indices of H such that $1 \le j_1 < j_2 < \dots < j_\ell \le m$.

We say a shortest u-v path P in G is a *critical path* with respect to H if $u \in X_{j_1} \cap H$ and $v \in X_{j_\ell} \cap H$. From the definition of a path partition, P intersects every bag X_t with $j_1 \le t \le j_\ell$. Hence, P intersects every critical bag of H.

▶ **Observation 16.** *For any critical path P with respect to H, $\mathsf{w}(\mathcal{P}[H \setminus P]) < \mathsf{w}(\mathcal{P}[H])$.*

Due to technical reasons, throughout the algorithm, we maintain a certain subsequence of the critical path P, called a *critical sequence*, which is defined as follows. For each critical index j of H, let $v_j \in X_j \cap H$ be the vertex that appears earliest in P along the direction from u to v. We may assume $v = v_{j_\ell}$ as otherwise, we can take the u-v_{j_ℓ} subpath of P. Then, $v_{j_1}, v_{j_2}, \dots, v_{j_\ell}$ are naturally ordered along the path P. We call $\langle v_{j_1}, v_{j_2}, \dots v_{j_\ell} \rangle$ a *critical sequence* of H with respect to P, and denote it by $\bar{P}[H]$. Note that this critical sequence $\bar{P}[H]$ is uniquely defined for a fixed H and P. See Figure 5(a) for an illustration. Since $\bar{P}[H]$ intersects every critical bag of H, we observe the following.

▶ **Observation 17.** *For a critical sequence $\bar{P}[H]$ of H, $\mathsf{w}(\mathcal{P}[H \setminus \bar{P}[H]]) < \mathsf{w}(\mathcal{P}[H])$.*

For a critical sequence $\bar{P} = \langle u, \dots, v \rangle$ of H with respect to P, and a strongly connected subgraph C of H, let x and y be the first and the last vertices of $\bar{P} \cap C$, respectively. We define the *surviving subsequence* of $\bar{P}$ with respect to C, denoted by $\bar{P}[C]$, as the consecutive subsequence of $\bar{P}[H]$ that starts at x and ends at y. Note that while $\bar{P}[C]$ may not be fully contained in C, the two endpoints x, y are contained in C. See Figure 5(b) for an illustration.

We define the *forward disatnce* and the *backward distnace* of C (with respect to P) as $d_G(x, y)$ and $d_G(y, x)$, respectively, and denote them by $f_{\bar{P}}(C)$ and $b_{\bar{P}}(C)$, respectively. Note that we analyze these distances with respect to the shortest paths in the original graph G, rather than the subgraph C. This is sufficient for our purpose since our goal is to bound the *weak diameter* of each SCC. We show that $\bar{P}[C]$ plays the role of a "critical" sequence for C. That is, the removal of $\bar{P}[C]$ from C reduces its path partition-width. Based on the following lemma, in the remainder of this section, we slightly abuse the notation and refer to $\bar{P}[C]$ as a *critical sequence* of C.

▶ **Lemma 18.** *Let C be a strongly connected subgraph of H and let $\bar{P}$ be a critical sequence of H with respect to a critical path P. Then, $\mathsf{w}(\mathcal{P}[C \setminus \bar{P}[C]]) < \mathsf{w}(\mathcal{P}[H])$.*

▶ **Observation 19.** *Let $\bar{P}$ be a critical sequence of H with respect to a critical path P. Then, for every strongly connected subgraph C of H, $\bar{P}[C]$ is a subsequence of a shortest path in G.*

Algorithm. We describe our algorithm Scissors-Carving $(G, \mathcal{P}, H, \bar{P}, \Delta)$. See Algorithm 2 for the pseudocode. It receives as input a directed graph G, a path partition $\mathcal{P}$ of G, a strongly connected subgraph H of G, a critical sequence $\bar{P}$ of H with respect to P, and a parameter $\Delta > 0$. Moreover, it outputs a random cutset S of $E(H)$ such that for every SCC C of $H \setminus S$, either $\mathsf{w}(\mathcal{P}[C]) < \mathsf{w}(\mathcal{P}[H])$ or the weak diameter of C is at most 12Δ.

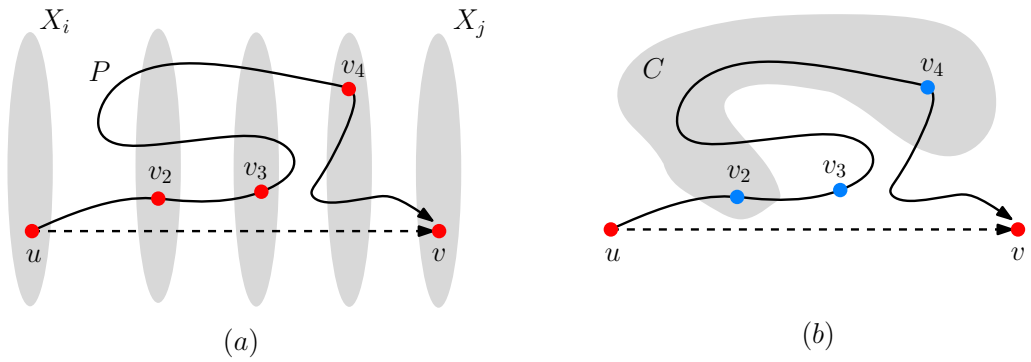


Figure 5 Illustration of the critical path and critical sequence of H. (a) The gray shaded region denotes the critical bags of H. A critical path P connects a vertex u in the first critical bag and a vertex v in the last critical bag. A critical sequence $\bar{P}$ of H with respect to P is $\langle u, v_2, v_3, v_4, v\rangle$. (b) For a subgraph C of H, the surviving subsequence $\bar{P}[C] = \langle v_2, v_3, v_4\rangle$ is depicted as blue points.

The algorithm is a recursive procedure with respect to H and $\bar{P}$. Depending on the forward distance $f_{\bar{P}}(H)$ and the backward distance $b_{\bar{P}}(H)$ of H, the procedure proceeds in one of three stages. The first stage is executed when $f_{\bar{P}}(H) = \tau > \Delta$. At this stage, we sample a random radius R uniformly at random from $[\frac{1}{2}\tau, \frac{2}{3}\tau]$. Then, we compute two balls, an out-ball $B_1 = B^+_G(u, R)$ and an in-ball $B_2 = B^-_G(v, R)$, compute a set S_1 of boundary edges of B_1 and B_2 that intersect H, i.e., $S_1 = H \cap (\delta^+(B_1) \cup \delta^-(B_2))$. Next, we recursively apply the algorithm to each SCC C of $H \setminus S_1$ which is fully contained in $B_1 \cup B_2$. In addition, we pass the surviving subsequence $\bar{P}[C]$ as the critical sequence to the next recursive call.

The second stage is executed when $f_{\bar{P}}(H) \leq \Delta$ and $b_{\bar{P}}(H) = \tau' > 8\Delta$. At this stage, we sample a random radius R uniformly at random from $[\frac{1}{2}\tau', \frac{13}{24}\tau']$. Then, we compute two balls, an in-ball $B_1 = B^-_G(u, R)$ and an out-ball $B_2 = B^+_G(v, R)$, compute a set S_1 of boundary edges of B_1 and B_2 that intersect H, i.e., $S_1 = H \cap (\delta^-(B_1) \cup \delta^+(B_2))$. Next, we recursively apply the algorithm to each SCC of $H \setminus S_1$ which is fully contained in $B_1 \cup B_2$. In addition, we pass the surviving subsequence $\bar{P}[C]$ to the next recursive call.

The third stage is executed when $f_{\bar{P}}(H) \leq \Delta$ and $b_{\bar{P}}(H) \leq 8\Delta$. We sample two random radii R_1, R_2 uniformly at random from $[\Delta, 2\Delta]$ and $[9\Delta, 10\Delta]$, respectively. Then, we compute two balls, an out-ball $B_1 = B^+_G(u, R_1)$ and an in-ball $B_2 = B^-_G(v, R_2)$, return $H \cap (\delta^+(B_1) \cup \delta^-(B_2))$.

Finally, we provide a crucial technical remark: As an extreme case, our algorithm immediately returns the empty set whenever $\bar{P}[C]$ is empty (lines 1-2). Intuitively, due to Lemma 18, $\bar{P}[C] = \emptyset$ implies $\mathsf{w}(\mathcal{P}[C]) < \mathsf{w}(\mathcal{P}[H])$, which satisfies our structural goal.

Algorithm 2 Scissors-Carving $(G, \mathcal{P}, H, \bar{P} = \langle u, \dots, v \rangle, \Delta)$.

if $\bar{P} = \emptyset$ **then**
  return $\emptyset$
$\tau, \tau', S_1, S_2 \leftarrow d_G(u,v), d_G(v,u), \emptyset, \emptyset$
/* **Stage 1. Reducing forward distance** */
if $\tau > \Delta$ **then**
  Sample R uniformly at random in the range $[\frac{1}{2}\tau, \frac{2}{3}\tau]$
  $B_1 \leftarrow B_G^+(u, R)$
  $B_2 \leftarrow B_G^-(v, R)$
  $S_1 \leftarrow H \cap (\delta^+(B_1) \cup \delta^-(B_2))$
  $\mathcal{C} \leftarrow$ SCCs of $H \setminus S_1$ that are fully contained in $B_1 \cup B_2$
  for $C \in \mathcal{C}$ **do**
    $S_2 \leftarrow S_2 \cup$ Scissors-Carving $(G, \mathcal{P}, C, \bar{P}[C], \Delta)$
  return $S_1 \cup S_2$
/* **Stage 2. Reducing backward distance** */
if $\tau \le \Delta$ and $\tau' > 8\Delta$ **then**
  Sample R uniformly at random in the range $[\frac{1}{2}\tau', \frac{13}{24}\tau']$
  $B_1 \leftarrow B_G^-(u, R)$
  $B_2 \leftarrow B_G^+(v, R)$
  $S_1 \leftarrow H \cap (\delta^-(B_1) \cup \delta^+(B_2))$
  $\mathcal{C} \leftarrow$ SCCs of $H \setminus S_1$ that are fully contained in $B_1 \cup B_2$
  for $C \in \mathcal{C}$ **do**
    $S_2 \leftarrow S_2 \cup$ Scissors-Carving $(G, \mathcal{P}, C, \bar{P}[C], \Delta)$
  return $S_1 \cup S_2$
/* **Stage 3. Bounding weak diameter** */
else
  Sample R_1 uniformly at random from $[\Delta, 2\Delta]$, and R_2 uniformly at random from $[9\Delta, 10\Delta]$
  $B_1 \leftarrow B_G^+(u, R_1)$
  $B_2 \leftarrow B_G^-(u, R_2)$
  return $H \cap (\delta^+(B_1) \cup \delta^-(B_2))$

5 Directed Sparsest Cut Problem for Bounded-Treewidth Digraphs

Directed non-bipartite sparsest cut problem. Let G be an edge-weighted directed graph with *capacities* $c(e)$ for all edges $e \in E(G)$. Let T be a set of terminal pairs $T = \{(s_1, t_1), (s_2, t_2), \dots, (s_\ell, t_\ell)\}$ of vertices. For each $i \in [\ell]$, let $\mathsf{dem}(i)$ be a non-negative real number, called the *demand*, for the terminal pair (s_i, t_i). A *cut* S of G is a subset $S \subseteq E(G)$ of edges. For a cut S, let $c(S) := \sum_{e \in S} c(e)$ be the *capacity* of S. Let $I_S \subseteq [\ell]$ be the set of integers i such that every s_i-t_i path (not necessarily a shortest path) contains at least one edge of S. We define the *demand* of S by $\mathsf{dem}(S) := \sum_{i \in I_S} \mathsf{dem}(i)$. The *sparsity* of S is defined by $c(S)/\mathsf{dem}(S)$. The goal of the directed non-bipartite sparsest cut problem is to find a cut S that minimizes the sparsity.

The integrality gap of the LP relaxation for this problem is called the directed multi-commodity flow cut-gap. For non-uniform demands, Hajiaghayi and Räcke showed an $O(\sqrt{n})$ upper bound for the flow-cut gap [27]. This has been improved to $\tilde{O}(n^{11/23})$ by Agarwal, Alon, and Charikar [4]. On the other hand, the current best-known lower bound is due to Chuzhoy and Khanna [14], who showed a $\tilde{\Omega}(n^{1/7})$ lower bound for the flow-cut gap.

Mémoli, Sidiropoulos, and Sridhar [40] showed that every n-vertex directed graph with treewidth tw admits an $(O(\mathsf{tw}\log n), \Delta)$-quasipartition for every parameter $\Delta > 0$. They used their quasipartition construction to show that the flow-cut gap is $O(\mathsf{tw}\log^2 n)$ for such graphs. We now present the main result of this section.

▶ **Theorem 4.** *Let G be an n-vertex directed graph of treewidth tw. The integrality gap (also called the flow-cut gap) of the Directed Non-Bipartite Sparsest-Cut LP relaxation on G is $O(\mathsf{tw}\log n)$.*

This improves upon the $\log n$ factor of the previously best-known bound from [40]. At a high level, we follow the procedure of [40], but provide a better analysis of their approach.

We briefly explain the approach of [40], which proved that the integrality gap is $O(\mathsf{tw}\log^2 n)$. First, they showed that for every $\Delta > 0$, G admits a (β, Δ)-quasipartition with $\beta = O(\mathsf{tw}\log n)$. Using this quasipartition, they proved that the quasimetric space with respect to the shortest path metric of G can be embedded into a convex combination of 0-1 quasimetric spaces (see Definition 21 and Lemma 22) with a distortion of $O(\beta\log n) = O(\mathsf{tw}\log^2 n)$. Finally, by using the known relationship between the convex combination of 0-1 quasimetric spaces and the integrality gap for the directed non-bipartite sparsest cut problem [13], they showed that the integrality gap is $O(\mathsf{tw}\log^2 n)$.

▶ **Definition 20.** *A quasimetric space $M = (X, d)$ is a function d over $X \times X$ that satisfies all but the symmetry axiom for a metric space. In other words, d satisfies the following properties:*

- *$d(u, u) = 0$ for all $u \in X$.*
- *(Triangle inequality) $d(u, v) \leq d(u, w) + d(w, v)$ for all $u, v, w \in X$.*

▶ **Definition 21.** *A quasimetric space $M = (X, d)$ is a 0-1 quasimetric space if every metric distance $d(u, v)$ is either 0 or 1.*

▶ **Lemma 22** (Lemma 16 of [40])**.** *Suppose a directed graph G admits a (β, Δ)-quasipartition for every $\Delta > 0$. Then there is an embedding of G into a convex combination of 0-1 quasimetric spaces with a distortion of $O(\beta\log n)$.*

5.1 Deep dive into the approach of [40]

In this section, we carefully review the previous approach of [40] step-by-step, and then describe our improvement. First, we analyze the $(O(\mathsf{tw}\log n), \Delta))$-quasipartition construction of [40].

The construction is achieved by a recursive approach. Given a tree decomposition $(T, \{B_t\}_{t\in T})$, we choose a central bag B_t, meaning that each subgraph of G induced by the maximal subtree of $T \setminus \{t\}$ contains at most $\frac{2}{3}|V(G)|$ vertices. We then sample a radius R uniformly at random from $[\frac{1}{2}\Delta, \Delta]$. For each vertex $v \in B_t$, we compute two balls, an out-ball $B_1 := B_G^+(v, R)$ and an in-ball $B_2 := B_G^-(v, R)$, add $\delta^+(B_1) \cup \delta^-(B_2)$ to the cutset S, and recursively apply this procedure to each subgraph induced by the connected components of $T \setminus \{t\}$. Note that any x-y path intersecting v such that $d_G(x, y) > 2\Delta$ must intersect $\delta^+(B_1) \cup \delta^-(B_2)$ because, due to the triangle inequality, we have that:

$$\max(d_G(x, v), d_G(v, y)) \geq \frac{1}{2}(d_G(x, v) + d_G(v, y)) \geq \frac{1}{2}d_G(x, y) > \Delta.$$

This implies that for every path (not necessarily a shortest path) connecting two vertices from different trees of $T \setminus \{t\}$, either it intersects S, or it satisfies $d_G(x, y) \leq 2\Delta$. This justifies the recursion: after the recursion terminates, we can conclude that for every x-y path in $G \setminus S$, we have $d_G(x, y) \leq 2\Delta$.

We briefly analyze the probability that a single edge belongs to the cutset, i.e., $\Pr[e \in S]$. Since B_t plays the role of a separator, each edge e is affected by at most one instance at each level of the recursion. Moreover, since B_t is a central bag, the recursion depth is $O(\log n)$. At a single level, e is affected by at most $2|B_t| = 2(\mathsf{tw}+1)$ balls. Hence, due to the union bound on the probability,

$$\Pr[e \in S] \leq O(\log n) \cdot O(\mathsf{tw}) \cdot \frac{w(e)}{\Delta} = O(\mathsf{tw} \log n) \cdot \frac{w(e)}{\Delta}.$$

Next, we analyze the second part which gives an embedding of a quasimetric space into a convex combination of 0-1 quasimetric spaces (Lemma 22). For simplicity, suppose every edge weight of G is a positive integer and $\max_{u,v}\{d_G(u,v)\} = 2^k$ for some integer $k = O(\log n)$. Then, for each scale $i \in \{0,1,\ldots,k\}$, we compute a $(\beta, 2^i)$-quasipartition $\mathcal{D}_i$ of G using the above procedure. Importantly, each quasipartition $\mathcal{D}_i$ is a distribution over a cutset $S_i \subset E(G)$. Using these, we compute a random quasipartition of G according to the following procedure. Let $c = 2^{k+1} - 1$. First, we select one $\mathcal{D}_i$ among $\{\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_k\}$ with probability $\frac{2^i}{c}$. Then, we pick a random cutset S from the chosen $\mathcal{D}_i$. This yields a distribution $\mathcal{D}$ over the cutsets.

Next, for each cutset S of $\mathcal{D}$, we define a 0-1 quasimetric space $(V(G), d_S)$ where $d_S(x,y) = 0$ if there is an x-y path in $G \setminus S$ and $d_S(x,y) = 1$ otherwise. This yields a distribution over 0-1 quasimetric spaces which can be expressed as a convex combination of 0-1 quasimetric spaces. Let $(V(G), d^*)$ be the resulting quasimetric space. Then, we have

$$d^*(u,v) = \Pr_{S\sim\mathcal{D}}[\text{there is no } u\text{-}v \text{ path in } G \setminus S] \tag{1}$$

$$= \sum_{i=0}^{k} \frac{2^i}{c} \Pr_{S_i\sim\mathcal{D}_i}[\text{there is no } u\text{-}v \text{ path in } G \setminus S_i] \tag{2}$$

$$\leq \sum_{i=0}^{k} \frac{2^i}{c} \beta \cdot \frac{d_G(u,v)}{2^i} \tag{3}$$

$$= \frac{(k+1)\cdot\beta}{c} d_G(u,v). \tag{4}$$

For the other direction, [40] showed that $d^*(u,v) \geq \frac{1}{c} \cdot d_G(u,v)$, yielding a distortion of $(k+1)\beta = O(\beta \log n)$.

5.2 Removing the extra $O(\log n)$ factor from Lemma 22

In this section, we provide a better analysis that improves the distortion by a factor of $O(\log n)$. We focus on the $O(\log n)$ additional factor arising from Lemma 22. This factor comes from combining the distortion contributions across $O(\log n)$ scales, where each scale contributes independently.

Our idea is to avoid this factor by considering all scales simultaneously. To formalize this idea, we revisit the construction in Section 5.1. For a fixed edge $e = (x,y)$ and a scale i, we compute a set $V(e) \subset V(G)$ of size $O(\mathsf{tw} \log n)$ as follows. For each recursive instance that contains e, we add the vertices in the central bag to $V(e)$. Since e is contained in $O(\log n)$ instances (from the construction in Section 5.1), we have $|V(e)| = O(\mathsf{tw} \log n)$. Note that $V(e)$ is independent of the scale i.

Now, we reinterpret the construction in Section 5.1 as follows. For every scale i, we sample a radius R_i uniformly at random from $[2^{i-1}, 2^i]$, and we add e to the cutset S_i if e is contained in one of the boundaries of the (in- and out-) balls of radius $R_i \in [2^{i-1}, 2^i]$ centered at some vertex in $V(e)$. This defines the quasipartition $\mathcal{D}_i$ over the cutsets. We prove the following useful lemma.

▶ **Lemma 23.** *Let $e = (x, y)$ be an edge and z be a vertex in $V(e)$. Then, there are at most $O(1)$ scales $i \in [k]$ satisfying the following conditions:*
(a) $\Pr[e \in \delta^+(B^+_G(z, R_i))] > 0$, *and*
(b) $2^i > w(e)$.

Proof. Let i be an index satisfying conditions $(a), (b)$. Note that e is contained in $\delta^+(B^+_G(z, R_i))$ if and only if $d_G(z, x) \leq R_i$ and $d_G(z, y) > R_i$. Subsequently, from condition (a), we have that:

$$\Pr_{R_i}[(d_G(z, x) \leq R_i) \cap (d_G(z, y) > R_i))] > 0.$$

Since R_i is sampled uniformly at random from $[2^{i-1}, 2^i]$, we have that:

$$d_G(z, x) < 2^i, \text{ and } d_G(z, y) \geq 2^{i-1}.$$

By the triangle inequality and condition (b), we obtain:

$$2^{i-1} \leq d_G(z, y) \leq d_G(z, x) + d_G(x, y) \leq d_G(z, x) + w(e) < 2^i + 2^i = 2^{i+1}.$$

Let j be the fixed integer such that $d_G(z, y) \in [2^j, 2^{j+1}]$. Then, $i \in \{j-1, j, j+1, j+2\}$. We conclude that there are at most 4 scales i satisfying both conditions $(a), (b)$. This completes the proof. ◀

Using exactly the same argument, we can show the following lemma.

▶ **Lemma 24.** *Let $e = (x, y)$ be an edge and z be a vertex in $V(e)$. Then, there are at most $O(1)$ scales $i \in [k]$ satisfying the following conditions:*
(a) $\Pr[e \in \delta^-(B^-_G(z, R_i))] > 0$, *and*
(b) $2^i > w(e)$.

For an edge e and a vertex $z \in V(e)$, let $I(z, e)$ be the set of indices i such that $\Pr[e \in \delta^+(B^+_G(z, R_i)) \cup \delta^-(B^-_G(z, R_i))] > 0$ and $2^i > w(e)$. Due to Lemmas 23 and 24, we have that:

$$|I(z, e)| = O(1). \tag{5}$$

Let $i \in I(z, e)$. Since R_i is chosen uniformly at random from $[2^{i-1}, 2^i]$, we have that:

$$\Pr_{R_i}[(d_G(z, x) \leq R_i) \cap (d_G(z, y) > R_i))] \leq O(1) \cdot \frac{w(e)}{2^i}. \tag{6}$$

Next, we analyze the probability $q := \Pr_{S \sim \mathcal{D}}[\text{there is no } u\text{-}v \text{ path in } G \backslash S]$. Our argument closely follows that of [40]. Let $\mathcal{P} = \{p_1, p_2, \ldots, p_\ell\}$ be the set of all u-v paths in G. For every $m \in [\ell]$, let X_m be the event that p_m contains at least one edge in S. Then, $q = \Pr[X_1 \wedge X_2 \wedge \ldots \wedge X_\ell] \leq \Pr[X_m]$ for all $m \in [\ell]$. Now let us consider a fixed *shortest path* $p_m = \langle u = v_1, v_2, \ldots, v_t = v \rangle \in \mathcal{P}$. For every $j \in [t-1]$, let Y_j be the event that $(v_j, v_{j+1}) \in S$. Then, $\Pr[X_m] = \Pr[Y_1 \vee Y_2 \vee \ldots \vee Y_{t-1}] \leq \sum_j \Pr[Y_j]$ due to the union bound of probability. In addition, let $e_j = (v_j, v_{j+1})$. Then,

$$\Pr[Y_j] = \Pr[e_j \in S] = \sum_{i=0}^{k} \frac{2^i}{c} \cdot \Pr[e_j \in S_i] \tag{7}$$

$$\leq \sum_{i=0}^{k} \frac{2^i}{c} \cdot \sum_{z \in V(e_j)} \Pr_{R_i}[(d_G(z, v_j) \leq R_i) \cap (d_G(z, v_{j+1}) > R_i))] \tag{8}$$

$$= \sum_{z \in V(e_j)} \sum_{i=0}^{k} \frac{2^i}{c} \cdot \Pr_{R_i}[(d_G(z, v_j) \leq R_i) \cap (d_G(z, v_{j+1}) > R_i))]. \tag{9}$$

We analyze the inner summation over i as follows. We split the sum over k scales at $i = \lfloor \log w(e_j) \rfloor$. Since $I(z, e_j) \subseteq \{i : 2^i > w(e_j)\}$, we can bound each part separately. Let $P_{i,z} := \Pr_{R_i}[(d_G(z, v_j) \le R_i) \cap (d_G(z, v_{j+1}) > R_i))]$. Then,

$$\begin{aligned}
&\sum_{i=0}^{k} \frac{2^i}{c} \cdot \Pr_{R_i}[(d_G(z, v_j) \le R_i) \cap (d_G(z, v_{j+1}) > R_i))] = \sum_{i=0}^{k} \frac{2^i}{c} \cdot P_{i,z} \\
&= \sum_{i=0}^{\lfloor \log w(e_j) \rfloor} \frac{2^i}{c} \cdot P_{i,z} + \sum_{i=\lfloor \log w(e_j) \rfloor + 1}^{k} \frac{2^i}{c} \cdot P_{i,z} \\
&\le \sum_{i=0}^{\lfloor \log w(e_j) \rfloor} \frac{2^i}{c} + \sum_{i \in I(z,e_j)} \frac{2^i}{c} \cdot P_{i,z} \\
&\overset{(6)}{\le} \frac{2 \cdot w(e_j)}{c} + \sum_{i \in I(z,e_j)} \frac{2^i}{c} \cdot (O(1) \cdot \frac{w(e_j)}{2^i}) \overset{(5)}{=} O(1) \cdot \frac{w(e_j)}{c}.
\end{aligned}$$

Returning to Equation (9), we obtain:

$$\begin{aligned}
\Pr[Y_j] &\le \sum_{z \in V(e_j)} \sum_{i=0}^{k} \frac{2^i}{c} \cdot \Pr_{R_i}[(d_G(z, v_j) \le R_i) \cap (d_G(z, v_{j+1}) > R_i))]. \\
&\le \sum_{z \in V(e_j)} O(1) \cdot \frac{w(e_j)}{c} = O(|V(e_j)|) \cdot \frac{w(e_j)}{c} = O(\mathsf{tw} \log n) \cdot \frac{w(e_j)}{c}.
\end{aligned}$$

Here, the last equality follows from the fact that $|V(e)| = O(\mathsf{tw} \log n)$ for every edge e. Since p_m is a shortest u-v path in G, we have that:

$$q \le \Pr[X_m] \le \sum_j \Pr[Y_j] \le O(\mathsf{tw} \log n) \cdot \sum_j \frac{w(e_j)}{c} = O(\mathsf{tw} \log n) \cdot \frac{d_G(u, v)}{c}. \tag{10}$$

Putting everything together, we show that the probability bound from Eq.(1) to Eq.(4) can be improved as follows:

$$d^*(u, v) = \Pr_{S \sim \mathcal{D}}[\text{there is no } u\text{-}v \text{ path in } G \setminus S] \overset{(10)}{\le} O(\mathsf{tw} \log n) \cdot \frac{d_G(u, v)}{c}.$$

This confirms the improved distortion guarantee for the algorithm of [40].

▶ **Lemma 25.** *Given a directed graph G having treewidth* tw*, there is an embedding of G into the convex combination of 0-1 quasimetric space with distortion $O(\mathsf{tw} \log n)$.*

▶ **Theorem 4.** *Let G be an n-vertex directed graph of treewidth* tw*. The integrality gap (also called the flow-cut gap) of the Directed Non-Bipartite Sparsest-Cut LP relaxation on G is $O(\mathsf{tw} \log n)$.*

References

1 I. Abraham, C. Gavoille, D. Malkhi, and U. Wieder. Strong-diameter decompositions of minor free graphs. *Theory of Computing Systems*, 47(4):837–855, 2010. doi:10.1007/s00224-010-9283-6.

2 Ittai Abraham, Arnold Filtser, Anupam Gupta, and Ofer Neiman. Metric embedding via shortest path decompositions. *SIAM J. Comput.*, 51(2):290–314, 2022. a priliminary version apperared in the proceedings of STOC 18. doi:10.1137/19m1296021.

3 Ittai Abraham, Cyril Gavoille, Anupam Gupta, Ofer Neiman, and Kunal Talwar. Cops, robbers, and threatening skeletons: Padded decomposition for minor-free graphs. *SIAM J. Comput.*, 48(3):1120–1145, 2019. preliminary version published in STOC 2014. doi:10.1137/17M1112406.

4 Amit Agarwal, Noga Alon, and Moses S Charikar. Improved approximation for directed cut problems. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 671–680, 2007. doi:10.1145/1250790.1250888.

5 Sepehr Assadi, Gary Hoppenworth, and Nicole Wein. Covering approximate shortest paths with dags. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, STOC '25, pages 2269–2280, New York, NY, USA, 2025. Association for Computing Machinery. doi:10.1145/3717823.3718298.

6 Yair Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 184–193, 1996. doi:10.1109/SFCS.1996.548477.

7 Piotr Berman, Arnab Bhattacharyya, Konstantin Makarychev, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Improved approximation for the directed spanner problem. In *International Colloquium on Automata, Languages, and Programming*, pages 1–12. Springer, 2011. doi:10.1007/978-3-642-22006-7_1.

8 Aaron Bernstein, Maximilian Probst Gutenberg, and Christian Wulff-Nilsen. Near-optimal decremental SSSP in dense weighted digraphs. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 1112–1122. IEEE, 2020. doi:10.1109/FOCS46700.2020.00107.

9 Aaron Bernstein, Danupon Nanongkai, and Christian Wulff-Nilsen. Negative-weight single-source shortest paths in near-linear time. *Commun. ACM*, 68(2):87–94, 2025. preliminary version published in FOCS 2022. doi:10.1145/3631536.

10 Guy E Blelloch, Anupam Gupta, Ioannis Koutis, Gary L Miller, Richard Peng, and Kanat Tangwongsan. Nearly-linear work parallel sdd solvers, low-diameter decomposition, and low-stretch subgraphs. *Theory of Computing Systems*, 55(3):521, 2014.

11 Greg Bodwin. New results on linear size distance preservers. *SIAM Journal on Computing*, 50(2):662–673, 2021. doi:10.1137/19M123662X.

12 Karl Bringmann, Nick Fischer, Bernhard Haeupler, and Rustam Latypov. Near-optimal directed low-diameter decompositions. In Keren Censor-Hillel, Fabrizio Grandoni, Joël Ouaknine, and Gabriele Puppis, editors, *52nd International Colloquium on Automata, Languages, and Programming, ICALP 2025, July 8-11, 2025, Aarhus, Denmark*, volume 334 of *LIPIcs*, pages 35:1–35:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.ICALP.2025.35.

13 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Directed metrics and directed graph partitioning problems. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 51–60, 2006. URL: http://dl.acm.org/citation.cfm?id=1109557.1109564.

14 Julia Chuzhoy and Sanjeev Khanna. Polynomial flow-cut gaps and hardness of directed cut problems. *Journal of the ACM (JACM)*, 56(2):1–28, 2009. doi:10.1145/1502793.1502795.

15 Jonathan Conroy and Arnold Filtser. How to protect yourself from threatening skeletons: Optimal padded decompositions for minor-free graphs. In Michal Koucký and Nikhil Bansal, editors, *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC 2025, Prague, Czechia, June 23-27, 2025*, pages 2281–2292. ACM, 2025. doi:10.1145/3717823.3718252.

16 Guoli Ding and Bogdan Oporowski. On tree-partitions of graphs. *Discrete Mathematics*, 149(1-3):45–58, 1996. doi:10.1016/0012-365X(94)00337-I.

17 Matthias Englert, Anupam Gupta, Robert Krauthgamer, Harald Räcke, Inbal Talgam-Cohen, and Kunal Talwar. Vertex sparsifiers: New results from old techniques. *SIAM J. Comput.*, 43(4):1239–1262, 2014. doi:10.1137/130908440.

18 Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, November 2004. preliminary version published in STOC 2003. doi:10.1016/j.jcss.2004.04.011.

19 Jittat Fakcharoenphol and Kunal Talwar. An improved decomposition theorem for graphs excluding a fixed minor. In *RANDOM-APPROX*, pages 36–46, 2003. doi:10.1007/978-3-540-45198-3_4.

20 Arnold Filtser. On strong diameter padded decompositions. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2019, September 20-22, 2019, Massachusetts Institute of Technology, Cambridge, MA, USA*, pages 6:1–6:21, 2019. doi:10.4230/LIPIcs.APPROX-RANDOM.2019.6.

21 Arnold Filtser. On sparse covers of minor free graphs, low dimensional metric embeddings, and other applications. In Oswin Aichholzer and Haitao Wang, editors, *41st International Symposium on Computational Geometry, SoCG 2025, Kanazawa, Japan, June 23-27, 2025*, LIPIcs, pages 49:1–49:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.SOCG.2025.49.

22 Arnold Filtser. Stochastic embedding of digraphs into dags. In Kasper Green Larsen and Barna Saha, editors, *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2026, Vancouver, BC, Canada, January 11-14, 2026*, pages 105–119. SIAM, 2026. doi:10.1137/1.9781611978971.5.

23 Arnold Filtser, Tobias Friedrich, Davis Issac, Nikhil Kumar, Hung Le, Nadym Mallek, and Ziena Zeif. Optimal padded decomposition for bounded treewidth graphs. *TheoretiCS*, 4, 2025. doi:10.46298/THEORETICS.25.22.

24 Arnold Filtser and Ofer Neiman. Light spanners for high dimensional norms via stochastic decompositions. *Algorithmica*, 84(10):2987–3007, 2022. doi:10.1007/S00453-022-00994-0.

25 Fedor V Fomin, Daniel Lokshtanov, Saket Saurabh, Michał Pilipczuk, and Marcin Wrochna. Fully polynomial-time parameterized computations for graphs and matrices of low treewidth. *ACM Transactions on Algorithms (TALG)*, 14(3):1–45, 2018. doi:10.1145/3186898.

26 Gregory Gutin, Mark Jones, and Magnus Wahlstrom. The mixed chinese postman problem parameterized by pathwidth and treedepth. *SIAM Journal on Discrete Mathematics*, 30(4):2177–2205, 2016. doi:10.1137/15M1034337.

27 Mohammad Taghi Hajiaghayi and Harald Räcke. An $O(n)$-approximation algorithm for directed sparsest cut. *Information Processing Letters*, 97(4):156–160, 2006. doi:10.1016/J.IPL.2005.10.005.

28 Sariel Har-Peled, Manor Mendel, and Dániel Oláh. Reliable spanners for metric spaces. *ACM Transactions on Algorithms*, 19(1):1–27, 2023. doi:10.1145/3563356.

29 Lior Kamma and Robert Krauthgamer. Metric decompositions of path-separable graphs. *Algorithmica*, 79(3):645–653, 2017. doi:10.1007/S00453-016-0213-0.

30 Mamadou M Kanté, Christophe Paul, and Dimitrios M Thilikos. A linear fixed parameter tractable algorithm for connected pathwidth. *SIAM Journal on Discrete Mathematics*, 36(1):411–435, 2022. doi:10.1137/20M1369361.

31 Ken-ichi Kawarabayashi and Anastasios Sidiropoulos. Embeddings of planar quasimetrics into directed ℓ_1 and polylogarithmic approximation for directed sparsest-cut. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 480–491. IEEE, 2022.

32 Philip Klein, Serge A. Plotkin, and Satish Rao. Excluded minors, network decomposition, and multicommodity flow. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, STOC '93, pages 682–690, New York, NY, USA, 1993. ACM. doi:10.1145/167088.167261.

33 Shimon Kogan and Merav Parter. New diameter-reducing shortcuts and directed hopsets: Breaking the $\sqrt{n}$ barrier. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2022*, pages 1326–1341. SIAM, 2022. doi:10.1137/1.9781611977073.55.

34 R Krauthgamer, JR Lee, M Mendel, and A Naor. Measured descent: a new embedding method for finite metrics. *Geometric & Functional Analysis GAFA*, 15(4):839–858, 2005.

35 Michael Lampis and Manolis Vasilakis. Structural parameterizations for two bounded degree problems revisited. *ACM Transactions on Computation Theory*, 16(3):1–51, 2024. doi:10.1145/3665156.

36 J. R. Lee and A. Naor. Extending lipschitz functions via random metric partitions. *Inventiones Mathematicae*, 160(1):59–95, 2005. doi:10.1007/s00222-004-0400-5.

37 James R Lee and Anastasios Sidiropoulos. On the geometry of graphs with a forbidden minor. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 245–254, 2009. doi:10.1145/1536414.1536450.

38 Frank Thomson Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *J. ACM*, 46(6):787–832, 1999. doi:10.1145/331524.331526.

39 Jason Li. Simpler and faster directed low-diameter decompositions, 2025. doi:10.48550/arXiv.2505.10244.

40 Facundo Mémoli, Anastasios Sidiropoulos, and Vijay Sridhar. Quasimetric embeddings and their applications. *Algorithmica*, 80(12):3803–3824, 2018. doi:10.1007/S00453-018-0415-8.

41 Satish Rao. Small distortion and volume preserving embeddings for planar and Euclidean metrics. In *Proceedings of the Fifteenth Annual Symposium on Computational Geometry, Miami Beach, Florida, USA, June 13-16, 1999*, pages 300–306, 1999. doi:10.1145/304893.304983.

42 Neil Robertson and Paul D Seymour. Graph minors. I. excluding a forest. *Journal of Combinatorial Theory, Series B*, 35(1):39–61, 1983. doi:10.1016/0095-8956(83)90079-5.

43 Ario Salmasi, Anastasios Sidiropoulos, and Vijay Sridhar. On constant multi-commodity flow-cut gaps for families of directed minor-free graphs. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 535–553. SIAM, 2019. doi:10.1137/1.9781611975482.34.

44 Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *Journal of the ACM (JACM)*, 51(6):993–1024, 2004. doi:10.1145/1039488.1039493.

Bi-Lipschitz Extensions and Outlier Embeddings into Trees

Shuchi Chawla
Department of Computer Science, University of Texas at Austin, TX, USA

Arnold Filtser
Department of Computer Science, Bar-Ilan University, Ramat Gan, Israel

Kristin Sheridan
Department of Computer Science, University of Texas at Austin, TX, USA

Yonatan Trachtenberg
Department of Computer Science, Bar-Ilan University, Ramat Gan, Israel

Abstract

We develop low distortion embeddings with outliers from arbitrary metrics into hierarchically separated trees (HSTs). In particular, we develop an efficient algorithm that for any $\epsilon > 0$, given an input metric (X, d), and a probabilistic embedding of all but k points from X into HSTs with distortion c, samples from a probabilistic embedding of all but $O(\frac{k}{\epsilon} \log k)$ points into HSTs that achieves distortion at most $(32 + \epsilon)c$.

Our results are based on two key technical components. First, we extend an algorithm of Munagala et al. [53] for minimizing the distortion of embeddings without outliers into HSTs to the setting with outliers. We combine this with new results on bi-Lipschitz extensions into trees and ℓ_1 space. In particular, we show that any probabilistic embedding into HSTs can be extended to k additional points with only a factor $O(\log k)$ of additional distortion. This bi-Lipschitz extension result utilizes a new probabilistic partitioning scheme that we call *onion partitioning*.

2012 ACM Subject Classification Theory of computation → Rounding techniques

Keywords and phrases metric embeddings, hierarchically separated trees, outliers

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.17

Category APPROX

Related Version *Full Version*: `https://arxiv.org/abs/2601.15470`

Funding *Shuchi Chawla*: This research was funded in part by NSF awards CCF-2217069 and CCF-2225259.
Arnold Filtser: This research was supported by the ISRAEL SCIENCE FOUNDATION (grant No. 1042/22).
Kristin Sheridan: This research was funded in part by NSF awards CCF-2217069 and CCF-2225259 and by JPMorgan Chase & Co. Any views or opinions expressed herein are solely those of the authors listed, and may differ from the views and opinions expressed by JPMorgan Chase & Co. or its affiliates. This material is not a product of the Research Department of J.P. Morgan Securities LLC. This material should not be construed as an individual recommendation for any particular client and is not intended as a recommendation of particular securities, financial instruments or strategies for a particular client. This material does not constitute a solicitation or offer in any jurisdiction.

1 Introduction

Low-distortion embeddings into trees are a fundamental algorithmic tool for reducing optimization problems on general graph metrics to the special case of tree metrics, where they are often easier to solve. The distortion of an embedding, defined as the maximum


© Shuchi Chawla, Arnold Filtser, Kristin Sheridan, and Yonatan Trachtenberg;
licensed under Creative Commons License CC-BY 4.0



multiplicative change in pairwise distances, directly determines the loss in performance incurred by such a reduction. A series of works [7, 12, 13, 37] showed that *probabilistic* embeddings can embed any n-point metric into hierarchically separated trees (HSTs) with expected distortion $O(\log n)$. While uniform distortion bounds are helpful in understanding the general effectiveness of using metric embeddings to solve a problem, from a practical perspective, we may ask how well a particular given metric (X, d) can embed into trees. Indeed, for some metrics the optimal distortion is $\Omega(\log n)$ but for others it may be much smaller, e.g. $O(1)$, providing better algorithmic results.

In this paper we investigate instance-optimal distortion for probabilistic embeddings into trees by exploring the role of outliers. For a finite metric (X, d), a (k, c)-outlier embedding is a map defined on "most of" X – that is on all but k vertices, which we designate as outliers – into another metric space Y such that the distortion over the non-outliers is bounded by c. Intuitively, we can view the outliers as noisy or adversarial data that may hurt the quality of the instance-optimal embedding from X into Y. The goal of the outlier embedding is to identify and remove these outliers upfront so as to obtain a much better embedding over the remaining points. Algorithmically, given a target distortion c, we want to find the smallest k such that the removal of k outliers enables embedding the rest of the metric with distortion at most c. In this paper, we develop probabilistic outlier embeddings into trees, providing a tradeoff between outlier set size and distortion. Our main result is as follows.

▶ **Theorem 1** (HSTs admit outlier embeddings). *Let (X, d) be any metric that admits a (k, c) probabilistic outlier embedding into HSTs. Then for any constant $\epsilon > 0$, there exists an algorithm that efficiently samples from a probabilistic embedding into HSTs of some $S \subseteq X$ with $|X \setminus S| \leq O(\frac{1}{\epsilon} k \log k)$ and expected distortion at most $(32 + \epsilon)c$.*

Technical contribution: improved bi-Lipschitz extensions

A key technical component in our construction is a bi-Lipschitz extension for embeddings of arbitrary finite metrics into trees. Given a finite metric (X, d) and a low distortion embedding α from a subset $S \subset X$ into a host space Y, a bi-Lipschitz extension extends the domain of α to the entire set X, while only modestly increasing the distortion of the embedding. We show that if the distortion of the original embedding α is c_α, it is possible to obtain a bi-Lipschitz extension with distortion at most $O(c_\alpha \log k)$ where $k = |X \setminus S|$. Formally, we achieve the following bounds, where the expansion factor L_{exp} is the multiplicative increase in pairwise distances beyond the expansion of the original embedding α, and the contraction factor L_{contr} is the multiplicative decrease beyond the contraction in α.

▶ **Theorem 2** (HSTs have probabilistic bi-Lipschitz extensions). *Let $\mathcal{H}$ be the class of all HSTs. For any subset $S \subseteq X$ with $k := |X \setminus S|$ and a probabilistic embedding $\mathcal{D}_S : S \hookrightarrow \mathcal{H}$, there exists a probabilistic bi-Lipschitz extension of $\mathcal{D}_S$, $\mathcal{D} : X \hookrightarrow \mathcal{H}$, with expansion factor $L_{exp} = O(\log k)$ and contraction factor $L_{contr} = 4$.*

Further, we can efficiently sample from the distribution $\mathcal{D}$ if we can efficiently sample from $\mathcal{D}_S$.

A connection between outlier embeddings and Lipschitz extensions was first established by Chawla and Sheridan [26] in the context of deterministic embeddings into normed metrics. They showed that deterministic embeddings into ℓ_2 admit a Lipschitz extension with expansion factor $O(\log k)$ (but potentially unbounded contraction), and this suffices to obtain a bicriteria approximation for outlier embeddings into ℓ_2. We extend this approach to embeddings into HSTs by combining it with an LP-rounding approximation algorithm of Munagala et al. [53] for instance-optimal probabilistic embedding into HSTs.

Probabilistic extension and bounded contraction

Extending the approach of [26] to HSTs poses two new technical challenges. First, while Chawla and Sheridan [26]'s construction can tolerate arbitrary contraction in the Lipschitz extension, in order to employ Munagala et al.'s LP relaxation for HST embeddings, we necessarily need to control contraction as well. The second challenge is applying the notion of Lipschitz extension to probabilistic embeddings. This is important because probabilistic tree embeddings can obtain exponentially smaller distortion than deterministic tree embeddings. Consider, in particular, a distribution $\mathcal{D}$ over embeddings from some subset S of a given metric (X, d) into a host space Y. Simply drawing an embedding α from $\mathcal{D}$ and extending it to X may result in distortion $\propto \mathbb{E}_{\alpha\sim\mathcal{D}}[\max_{u,v\in X}(d_\alpha(u,v)/d(u,v))]$. This can be much larger than the desired distortion $\max_{u,v\in X}(\mathbb{E}_\alpha[d_\alpha(u,v)]/d(u,v))$.

We resolve both challenges by constructing a bi-Lipschitz extension through a sequence of "merge" operations, each of which combines two embeddings over domains that intersect in a single point. By carefully aligning and stitching each such pair of embeddings, we ensure strong bounds on both contraction and expansion.

Improved expansion via "onion-like" partitions

Chawla and Sheridan's Lipschitz extension for ℓ_2 employs a probabilistic partitioning procedure over the outliers $X \setminus S$ similar to that of Calinescu, Karloff, and Rabani [22], and attaches an embedding of each component of the partition to its closest point in S (that we call the *anchor point*). This approach combined with the merge operation described above yields a *bi-Lipschitz* extension for HSTs with expansion factor $O(\log^2 k)$. We develop a new technique to obtain a quadratic improvement over this factor, leading to Theorem 2. Our key observation is that the largest distortion is suffered by pairs that are separated into different components by the CKR partitioning procedure – one logarithmic factor arises from the probability of separation, and the second arises from the distortion between the points and their respective anchor points.

To improve upon this distortion, we construct a probabilistic partitioning where each component has an "onion-shell" shape – all points are roughly equidistant from the anchor point. We then employ an embedding of each component that preserves distances to the anchor point within a constant factor (while potentially inflating other distances by a logarithmic factor). This allows us to bound the distortion between any pair of points by $O(\log k)$ times the distortion of the embedding over S.

Our improved partitioning approach also applies to ℓ_1, providing a bi-Lipschitz extension with expansion factor $O(\log k)$. [26] previously obtained a bi-Lipschitz extension for ℓ_1 with expansion factor $O(\frac{1}{c_S}\log^2 k + \log k)$, where c_S is the distortion of the embedding of S into ℓ_1; this can be a logarithmic factor worse than the bound we achieve when c_S is $O(1)$.

▶ **Theorem 3** (ℓ_1 has bi-Lipschitz extensions, informal)**.** *Deterministic ℓ_1 embeddings have deterministic bi-Lipschitz extensions with expansion factor $O(\log k)$ and contraction factor $O(1)$.*

Related work

Embedding into HSTs has been studied primarily with the objective of minimizing expected distortion. See e.g. [7, 12, 13, 24, 37, 14, 46]. In our work, we will use as a subroutine the algorithm of Fakcharoenphol et al. [37], which for any n point metric samples from an $O(\log n)$ distortion probabilistic embedding into HSTs. *Instance-optimal* distortion has been studied for target spaces such as constant dimensional Euclidean space [10, 51, 30, 31, 58],

the line [9, 51, 38, 57], trees [11, 27], and ultrametrics [6, 53]. The goal in such embeddings is to minimize the distortion needed for the given *specific* (X, d), which in some cases may be smaller than the best bound achievable for all metrics of that size.

The notion of outlier embeddings was initially defined by Sidiropolous et al. [59] when they studied deterministic embeddings into ultrametrics, trees, and constant-dimensional ℓ_2 space. They consider approximating *deterministic, isometric* outlier embeddings (where the distortion c is 1) and bicriteria approximations with additive distortion. The structural properties used for isometric approximations don't apply to non-isometric embeddings, and techniques for bounding additive distortion tend to be quite different from those for multiplicative distortion, the focus of this work. Chubarian and Sidiropolous [29] continued the work on outlier embeddings and studied non-isometric outlier embeddings of unweighted graph metrics into the line. Later, Chawla and Sheridan [26] studied deterministic outlier embeddings into ℓ_2 space with unbounded dimension.

Several related notions of distortion were previously studied. One notion is *embeddings with slack*, in which the goal is to minimize the number of pairs that are distorted by more than some target factor [1, 23, 47]. An even stronger notion is that of *scaling distortion*, where for every $\varepsilon > 0$, at most ε fraction of the pairs suffer from distortion $\geq f(\varepsilon)$ (for some function f), see [45, 3, 4, 16]. Another notion is *terminal distortion*, where instead of an outlier set, there is a subset of terminals $K \subseteq X$, and the goal is only to preserve distance pairs involving a terminal $K \times X$ [32, 49, 56, 28]. A stronger notion is *prioritized distortion* [33, 34, 39] where there is a priority ordering over the vertices, and the distortion guarantee is w.r.t. the priorities. Another notion is that of local embeddings [2, 8, 25] where the goal is to minimize the distortion from each point to its k nearest neighbors.

Outlier embeddings are also closely related to metric Ramsey theory [21, 17, 19, 18, 52, 55, 20, 5, 15, 41, 42]. Given inputs n and c, metric Ramsey theory asks: What is the largest m such that *all* metrics of size n have a submetric of size m that embeds into some target space with distortion at most c? In this paper we are primarily focused on *instance-optimal* solutions rather than universal bounds that apply to all metrics, and we seek to approximate the size of the outlier set rather than the size of the non-outlier set.

Many other types of embedding compositions or extensions have been considered. One particular example is the notion of a Lipschitz extension [44, 48, 54], which seeks to leave non-outliers unaltered from the original embedding and ensure the expansion (but not necessarily the contraction) of every pair of nodes is bounded. Given an embedding on $S \subseteq X$, Calinescu, Karloff, and Rabani [22] develop a partitioning algorithm for $X \setminus S$ that implicitly defines Lipschitz extensions with a contraction factor at most $O(\log |S|)$ for all metric spaces. Chawla and Sheridan [26] define essentially the same partitioning algorithm, except that in their algorithm, [22] pick a uniformly random permutation over the set S, but [26] take a random permutation over the set of $X \setminus S$'s closest neighbors in S. The resulting Lipschitz extension factor is $O(\log |X \setminus S|)$. We will permute over the same set as [26], but we follow the cleaner analysis of [22]. Another related notion is that of 0-extension [22, 36, 35, 40] where one seeks a stochastic contraction $f : X \to K$ from the metric point set to a subset K of terminals while minimizing $\max_{x,y \in X} \mathbb{E}\left[\frac{d_X(f(x),f(y))}{d_X(x,y)}\right]$.

2 Definitions and results

Embeddings and distortion

A metric space is a set of points X with distances $d_X(\cdot, \cdot)$ between them that obey the triangle inequality. That is, for all $x, y, z \in X$, $d_X(x, z) \leq d_X(x, y) + d_X(y, z)$. We assume without loss of generality (via scaling) that in the metrics we consider, the minimum

distance between distinct pairs of points is at least 1. For subsets $S, T \subseteq X$, denote $d(S,T) := \min_{s\in S, t\in T}\{d(s,t)\}$ If $\alpha : X \to Y$ is a map from (X, d_X) to (Y, d_Y), we write $d_\alpha(x,y) := d_Y(\alpha(x), \alpha(y))$ to denote the distance between the images of two points $x, y \in X$ under this map.

$\mathcal{D}$ is a probabilistic embedding of (X, d_X) into $\mathcal{Y}$ if for each function α in the support of $\mathcal{D}$, $\alpha : X \to Y(\alpha)$ for some $(Y(\alpha), d_{Y(\alpha)}) \in \mathcal{Y}$. This embedding has expected *expansion* $c_{exp} \geq 1$, *contraction* $c_{contr} \geq 1$, and expected *distortion* $c_{contr} \cdot c_{exp}$ if for all $x, y \in X$ we have:

$$d_\alpha(x,y) \geq \frac{1}{c_{contr}} \cdot d(x,y) \qquad \forall \alpha \in \text{support}(\mathcal{D})$$
$$\mathop{\mathbb{E}}_{\alpha\sim\mathcal{D}}[d_\alpha(x,y)] \leq c_{exp} \cdot d(x,y).$$

Observe that *every* embedding in the support of $\mathcal{D}$ has contraction at most c_{contr}, but pairwise expansion (and thus distortion) is bounded in expectation over the randomness in $\mathcal{D}$. If the support of $\mathcal{D}$ contains a single embedding, it is a *deterministic* embedding. Further, an embedding is *expanding* or *non-contracting* if it has contraction 1.

We use the notation $\alpha : X \xhookrightarrow{c_{exp}, c_{contr}} Y$ to denote a deterministic or probabilistic embedding with expansion c_{exp} and contraction c_{contr} from (X, d_X) into (Y, d_Y). If c_{contr} is not specified, the embedding is expanding.

Bi-Lipschitz extensions

▶ **Definition 4** (Bi-Lipschitz extension). *Let (X, d) be a metric, $S \subseteq X$, and $\alpha_S : S \xhookrightarrow{c_S, c'_S} Y$ for some parameters c'_S, c_S. Then an embedding $\alpha : X \to Y$ is a (strong)* bi-Lipschitz extension *of α_S with extension factors L_{exp}, L_{contr} if* [1]

$$\text{for all } u \in S, \qquad \alpha(u) = \alpha_S(u), \tag{1}$$
$$\text{for all } u, v \in X \qquad d_\alpha(u,v) \geq \frac{1}{c'_S \cdot L_{contr}} d(u,v) \tag{2}$$
$$\text{and, for all } u, v \in X, \qquad d_\alpha(u,v) \leq L_{exp} \cdot c_S \cdot d(u,v). \tag{3}$$

If Y is a normed vector space, for Constraint 1 we require only that the restriction of α to the subspace spanned by S matches the embedding α_S. [2]

Further, α is a weak bi-Lipschitz extension *with extension factors L_{exp}, L_{contr} if Constraint 1 is replaced with the following weaker constraint:*

$$\text{for all } u, v \in S, \qquad d_\alpha(u,v) = d_{\alpha_S}(u,v).$$

In this paper, we discuss bi-Lipschitz extensions in the probabilistic setting. In particular, we model a probabilistic bi-Lipschitz extension on the weak version of a deterministic bi-Lipschitz extension, requiring that the *joint distribution* of distances between pairs of nodes in s is the same under the extended embedding.

[1] Note that L_{exp}, L_{contr} could be factors that depend on X and S.

[2] Notably, to get non-trivial extensions we must allow α to "extend" the dimension of α_S in some cases. To see this, consider an isometric (distortion 1) embedding of one or two points of a length n cycle into the real line. Any bi-Lipschitz extension that also embeds into the real line must have $L_{ext} \cdot L_{contr} = \Omega(n)$, as $\Omega(n)$ distortion is required to embed the cycle into a line.

▶ **Definition 5** (Probabilistic bi-Lipschitz extension). *Let (X, d) be a metric, $S \subseteq X$, and $\mathcal{D}_S \xrightarrow{c_S, c'_S} \mathcal{Y}$ for some parameters c_S, c'_S. Then a probabilistic embedding $\mathcal{D} : X \to \mathcal{Y}$ is a* probabilistic bi-Lipschitz extension *of α_S with extension factors L_{exp}, L_{contr} if*

$$(d_\alpha(x,y))_{x,y \in S \times S} \text{ has the same joint distribution for } \alpha \sim \mathcal{D} \text{ and } \alpha \sim \mathcal{D}_S \tag{4}$$

$$\text{for all } x, y \in X, \text{ and for all } \alpha \in \operatorname{support}(\mathcal{D}), \qquad d_\alpha(x,y) \geq \frac{1}{L_{contr} \cdot c'_S} \cdot d(x,y) \tag{5}$$

$$\text{and, for all } x, y \in X, \qquad \mathop{\mathbb{E}}_{\alpha \sim \mathcal{D}}[d_\alpha(x,y)] \leq L_{exp} \cdot c_S \cdot d(x,y). \tag{6}$$

Outlier embeddings

If $\mathcal{D}$ is a probabilistic embedding from (X, d_X) into $\mathcal{Y}$ and $K \subseteq X$, define $\mathcal{D}_{X \setminus K}$ to be the distribution identical to $\mathcal{D}$ but with all functions α in the support restricted to the set $X \setminus K$ (i.e. we can sample from $\mathcal{D}_{X \setminus K}$ by sampling from $\mathcal{D}$ and restricting the domain of the function drawn).

▶ **Definition 6.** *A probabilistic embedding $\mathcal{D}$ of (X, d_X) into $\mathcal{Y}$ is a* probabilistic (k, c)-outlier embedding *if there exists $K \subseteq X$ such that $|K| \leq k$ and $\mathcal{D}_{X \setminus K}$ is a probabilistic embedding of $(X \setminus K, d_X)$ into $\mathcal{Y}$ with distortion c.*

Note that in Definition 6, the subset K is chosen *deterministically* (not sampled). The distances from K to other nodes (and each other) are ignored. The randomness is only on the embedding for the remaining nodes in $X \setminus K$.

Hierarchically Separated Trees (HSTs), ultrametrics, and ℓ_p

▶ **Definition 7.** *A metric (X, d) is a* β-hierarchically separated tree (β-HST) *if there is a map from X to the leaves of a tree such that the following hold:*

- *Each node u in the tree is associated with ("labeled" with) some value η_u, and $\eta_v = \beta \cdot \eta_u$ whenever u is a child of v and u is not a leaf.*[3] *Further, for all leaves v, $\eta_v = 0$, and for nodes v that are parents of leaves, $\eta(v) = 1$.*
- *For two leaves $x, y \in X$ with least common ancestor z, $d(x, y) = \eta_z$.*

For simplicity, we will primarily focus on 2-HSTs, and in referring to HSTs, we mean 2-HSTs unless otherwise noted. We use $\mathcal{H}$ to denote the set of all 2-HSTs. Note that focusing on β-HSTs for a different constant β at worst changes results by a constant factor.

▶ **Definition 8.** *An* ultrametric *is a metric (X, d) that fulfills a strong triangle inequality guarantee: for all $x, y, z \in X$, $d(x, y) \leq \max\{ d(x, z), d(z, y) \}$.*

HSTs form a subset of the set of *ultrametrics*. Further, any ultrametric embeds into an HST with distortion at most 2 [17]. We will use $\mathcal{U}$ to denote the set of all ultrametrics.

▶ **Definition 9** (ℓ_p space). *ℓ_p in dimension δ, denoted ℓ_p^δ, is $(\mathbb{R}^\delta, d_{\ell_p})$ where for $x, y \in \mathbb{R}^\delta$, d_{ℓ_p} is defined as*

$$d_{\ell_p}(x,y) := \left(\sum_{i=1}^{\delta} |x_i - y_i|^p \right)^{1/p}.$$

We let ℓ_p space refer to the set of all spaces ℓ_p^δ for $\delta \in [1, \infty)$.

[3] In this paper, we will use the term β-HSTs to refer to what other sources sometimes call exact β-HSTs. These sources typically define non-exact β-HSTs as only requiring $\eta_v \geq \beta \cdot \eta_u$.

Main results

Our primary results are for probabilistic bi-Lipschitz extensions of embeddings into HSTs and deterministic bi-Lipschitz extensions of embeddings into ℓ_1 space.

▶ **Theorem 2** (HSTs have probabilistic bi-Lipschitz extensions). *Let $\mathcal{H}$ be the class of all HSTs. For any subset $S \subseteq X$ with $k := |X \setminus S|$ and a probabilistic embedding $\mathcal{D}_S : S \hookrightarrow \mathcal{H}$, there exists a probabilistic bi-Lipschitz extension of $\mathcal{D}_S$, $\mathcal{D} : X \hookrightarrow \mathcal{H}$, with expansion factor $L_{exp} = O(\log k)$ and contraction factor $L_{contr} = 4$.*

Further, we can efficiently sample from the distribution $\mathcal{D}$ if we can efficiently sample from $\mathcal{D}_S$.

We can also argue for the existence of deterministic extensions for ℓ_1 embeddings, though we do not know how to find such deterministic embeddings efficiently.

▶ **Theorem 3** (ℓ_1 has bi-Lipschitz extensions). *For any subset $S \subseteq X$ with an embedding $\alpha_S : S \xhookrightarrow{c_S, c'_S} \ell_1$, there exists*

1. *a deterministic* weak *bi-Lipschitz extension of α_S, $\alpha : X \hookrightarrow \ell_1$ with extension factors $L_{exp} = O(\log k)$ and $L_{contr} = 1$, where $k := |X \setminus S|$ and*
2. *a deterministic strong bi-Lipschitz extension with $L_{exp} = O(\log k)$ and $L_{contr} = 7$.*

Our bi-Lipschitz extensions into ℓ_1 are asymptotically optimal. Some metrics on n points (namely expander graph metrics) require $\Omega(\log n)$ distortion for embedding into ℓ_1 [50]. We can always "extend" a distortion 1 embedding on one or two points to an embedding on an expander graph metric, obtaining an embedding of distortion at most $L_{exp} \cdot L_{contr}$. Thus, we must have $L_{exp} \cdot L_{contr} = \Omega(\log k)$. Likewise, some metrics require $\Omega(\log n)$ distortion to embed probabilistically into HSTs [12], so our HST results are also asymptotically optimal.

Our results for probabilistic bi-Lipschitz extensions into HSTs allow us to obtain a bicriteria approximation for probabilistic outlier embeddings into HSTs, which we discussed in the introduction.

▶ **Theorem 1** (HSTs admit outlier embeddings). *Let (X, d) be any metric that admits a (k, c) probabilistic outlier embedding into HSTs. Then for any constant $\epsilon > 0$, there exists an algorithm that efficiently samples from a probabilistic embedding into HSTs of some $S \subseteq X$ with $|X \setminus S| \leq O(\frac{1}{\epsilon} k \log k)$ and expected distortion at most $(32 + \epsilon)c$.*

In Section 6, we expand our results to the weighted outlier setting, and we use a modification of the LP from Munagala et al. [53] to prove Theorem 1. Our LP-based techniques also allow us to obtain the following result for the weighted variant of the problem.[4]

▶ **Corollary 10.** *Given a metric (X, d), node weights w, and target distortion c, let w^* be the optimal outlier set cost for probabilistically embedding (X, d) into HSTs with c distortion and let k be the size of such an optimal outlier set. Then for any constant $\epsilon > 0$, there exists an algorithm that efficiently samples from a probabilistic embedding into HSTs of some $S \subseteq X$ with $w(X \setminus S) \leq O(\frac{1}{\epsilon} w^* \log k)$ and expected distortion at most $(32 + \epsilon)c$.*

[4] In the weighted outlier set problem we are given weights w_i on all nodes i in the metric space. The cost of a subset K of the metric space is $w(K) := \sum_{i \in K} w_i$, and the goal is to find an outlier embedding with some target distortion such that the cost of the outlier set is minimized.

3 Components of a bi-Lipschitz extension

The bi-Lipschitz extensions we discuss in this paper have three primary components. In particular, given a probabilistic embedding $\mathcal{D}_S : S \to \mathcal{Y}$ for $S \subseteq X$, we develop bi-Lipschitz extensions for $\mathcal{D}_S$ when the relevant spaces have certain properties. Namely, the space X should admit what we refer to as "onion partitions" and the space $\mathcal{Y}$ should admit "bounded-diameter embeddings" and "perfect merge functions." We show in Section 4 that onion partitions always exist, so our bi-Lipschitz extensions depend on the existence of bounded-diameter embeddings and perfect merge functions for the space into which we are embedding.

At a high level, we will use the onion partitions to find clusters of the set $K = X \setminus S$ and assign an "anchor point" in S to each such cluster. Then we find a bounded-diameter embedding on each cluster with its anchor point, and we "merge" each of these embeddings with the original embedding on S using a perfect merge function.

3.1 Onion partitions

We begin by defining what we call onion partitions. Given a metric (X, d) and a subset $S \subseteq X$ with $K = X \setminus S$, an onion partition of K is a partition $\mathcal{K} = \{ K_i \}$ of K such that each K_i has an assigned "anchor point" from S. For any point $x \in K_i$, we refer to its cluster's assigned point in S as the anchor point of x. The partition should satisfy three properties: (1) for any $x, y \in K$, the probability that x and y are separated in the partition is proportional to $d(x, y)/d(\{ x, y \}, S)$; (2) for each $x \in K$, the distance between x and its anchor point is (deterministically) similar to $d(x, S)$; and (3) for any cluster $K_i \in \mathcal{K}$, the distance between $x \in K_i$ and its assigned anchor point is (deterministically) not too much smaller than the diameter of the cluster K_i itself.

▶ **Definition 11** (Onion partitions). *Let (X, d) be a metric and let $S \subseteq X$ with $K := X \setminus S$. Then an* onion partition *of K is a joint distribution over $(\mathcal{K} = \{ K_i \}_{i \in \mathbb{N}}, r : \mathcal{K} \to S)$ such that $\mathcal{K}$ is a partition of K.*

For any $(\mathcal{K}, r)$ in the support of the distribution and $K_i \in \mathcal{K}$, we call $r(K_i)$ the anchor point *of K_i. We extend the domain of r to points in K: for $x \in K$, $r(x)$ denotes the anchor point assigned to the component containing x in the random partition $(\mathcal{K}, r)$. Likewise, let Δ_{K_i} denote the diameter of a component K_i, and Δ_x denote the (random) diameter of the component containing x in the random partition $(\mathcal{K}, r)$. Additionally, for any fixed $(\mathcal{K}, r)$ we let K_i^+ denote the set $K_i \cup \{ r(K_i) \}$ for each $K_i \in \mathcal{K}$.*

An onion partition has separation parameter $\rho > 0$, closeness $b > 1$, *and* band thickness $\gamma > 1$ *if it meets the following three criteria:*

1. ***Low separation probability.*** *For any $x, y \in K$, the probability that x and y are in different components of the partition is upper bounded by $\rho \cdot \frac{d(x,y)}{d(\{ x,y \},S)}$*
2. ***An anchor point is a "close" point in S.*** *For any $x \in K$ and any $(\mathcal{K}, r)$ in the support of the distribution, $d(x, r(x)) \leq b \cdot d(x, S)$ with probability 1.*
3. ***Cluster points are arranged in a thin band around their anchor point.*** *For any $x \in K$ and any $(\mathcal{K}, r)$ in the support of the distribution, $d(x, r(x)) \geq \Delta_x / \gamma$ with probability 1.*

Figure 1 gives a visual depiction of the third property of onion partitions, and in Section 4, we show the following theorem about their existence.

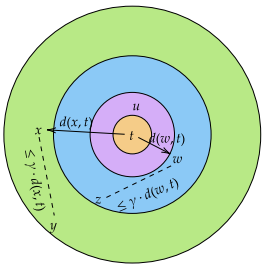


Figure 1 This figure depicts the third property of onion partitions. In particular, for any point $t \in S$, we think of the clusters with anchor point t as rings around t (like the rings of an onion). For any $x, y \in K$, if x and y are in the same cluster with anchor point $r(x) = r(y) = t$, then the third criterion of onion partitions states that they must be a distance at most $\gamma \cdot d(x,t)$ apart.

▶ **Theorem 12** (Onion partitions). *For any metric (X,d) and any $S \subseteq X$, there exists an onion partition on $K = X \setminus S$ with separation parameter $O(\log k)$, closeness 2, and band thickness 6, where $k = |K|$.*

3.2 Bounded-diameter embeddings

The second component of our extension method involves what we call bounded-diameter embeddings. In particular, we consider expanding embeddings that are parameterized not only by their worst expected expansion for any pair of nodes, but also by their expected expansion of the *diameter* of the original metric. That is, a bounded-diameter embedding can place a smaller upper bound on the expected expansion for pairs of nodes that are relatively far apart in the original metric than it does for other pairs of nodes. We define this formally in Definition 13.

▶ **Definition 13** (Bounded diameter embedding). *A probabilistic embedding $\mathcal{D} : X \to \mathcal{Y}$ is an expanding* bounded-diameter embedding *with expansion c and* diameter expansion *c_Δ if the following conditions hold:*

$$d_\alpha(x,y) \geq d(x,y) \qquad \forall x,y \in X, \forall \alpha \in \text{support}(\mathcal{D})$$
$$\mathop{\mathbb{E}}_{\alpha \sim \mathcal{D}}[d_\alpha(x,y)] \leq c \cdot d(x,y) \qquad \forall x,y \in X$$
$$\mathop{\mathbb{E}}_{\alpha \sim \mathcal{D}}[d_\alpha(x,y)] \leq c_\Delta \cdot \max_{x',y' \in X} \{ d(x',y') \} \qquad \forall x,y \in X$$

Further, we say $\mathcal{Y}$ admits bounded-diameter embeddings with expansion c_k and diameter expansion *c_Δ if for all metrics (X,d) of size k, there is a bounded-diameter embedding of X into $\mathcal{Y}$ with the given parameters.*

Notably, if all metrics of size k can embed into $\mathcal{Y}$ with expected expansion at most c_k, then $\mathcal{Y}$ admits bounded-diameter embeddings with expansion and diameter expansion equal to c_k. However, in the settings we consider here (namely HSTs and ℓ_1 space), we will consider bounded-diameter embeddings where c_Δ is tighter than c_k. In this paper, we will use FRT embeddings [37] as our bounded diameter embeddings.

▶ **Theorem 14** ([37]). *For any metric (X,d) with $|X| = n$, there exists a probabilistic expanding bounded diameter embedding (an FRT embedding) of X into HSTs with expansion $O(\log k)$ and diameter expansion 2.*

In fact, for HST embeddings, we can assume without loss of generality an even stronger property than that required by Definition 13: For any embedding α in the support of an HST embedding on X, the diameter of the image of α is at most twice the diameter of X. To see this, let Δ be the diameter of X and consider an HST embedding with diameter larger than $2^{\lceil \log \Delta \rceil}$. The nodes of the HST at level $2^{\lceil \log \Delta \rceil}$ or higher can all be merged into a single node without incurring any new contraction between any pair of points, as their new distance under this contraction is the minimum of their old distance and $2^{\lceil \log \Delta \rceil}$.

Further, trees embed isometrically into ℓ_1, so we immediately get the following corollary.

▶ **Corollary 15.** *For any metric (X, d) with $|X| = n$, there exists a probabilistic expanding bounded diameter embedding (an FRT embedding) of X into ℓ_1 with expansion $O(\log k)$ and diameter expansion 2.*

3.3 Perfect merge functions

In the final step of our extension method, we "merge" the embedding α_S on $S \subseteq X$ with bounded-diameter embeddings on the clusters of our onion partition. In particular, a *merge function* is a function whose input is two embeddings whose domains overlap at a single point and whose output is a new embedding on the union of their domains. For our purposes, we want to ensure some of the necessary distance properties are maintained. In particular, for any pair of nodes that appeared in the domain of the same input embedding, the distance under the new embedding should be the same as under the original one. Additionally, we want to ensure that no pair of nodes is allowed to be "too close" in the new embedding. This extra requirement will allow us to bound contraction in our final bi-Lipschitz extension.

▶ **Definition 16** (Perfect merge). *A procedure `Merge` is a* perfect merge function *for embedding X into $\mathcal{Y}$ with* contraction factor *η if for any $Z_1, Z_2 \subseteq X$ with $Z_1 \cap Z_2 = \{v\}$ and any deterministic embeddings $\alpha_1 : Z_1 \to Y_1$, $\alpha_2 : Z_2 \to Y_2$ for $Y_1, Y_2 \in \mathcal{Y}$, it produces an embedding $\alpha : Z_1 \cup Z_2 \to Y$ for some $Y \in \mathcal{Y}$ such that:*

1. *For all $x, y \in Z_1$, $d_\alpha(x, y) = d_{\alpha_1}(x, y)$.*
2. *For all $x, y \in Z_2$, $d_\alpha(x, y) = d_{\alpha_2}(x, y)$.*
3. *For $x \in Z_1$ and $y \in Z_2$, we have $d_\alpha(x, y) \geq \frac{d_{\alpha_1}(x,v) + d_{\alpha_2}(v,y)}{\eta}$.*

We say $\mathcal{Y}$ admits perfect merge functions if there is a perfect merge function for any metric X.

In Section 5, we prove the following existence results about perfect merge functions for HSTs and ℓ_1.

▶ **Lemma 17.** *HSTs admit a perfect merge function with contraction factor 2.*

▶ **Lemma 18.** *ℓ_1 admits a perfect merge function with contraction factor 1.*

Putting the components together

With our three components defined, we now put them together to construct bi-Lipschitz extensions for amenable target spaces $\mathcal{Y}$. In particular, given a metric (X, d) with subset $S \subseteq X$ and $\mathcal{D}_S : S \to \mathcal{Y}$, our extension proceeds in the following manner:

1. Sample an embedding on α_S from $\mathcal{D}_S$.
2. Sample an onion partition $(\mathcal{K}, r)$ of $K = X \setminus S$.
3. For each $K_i \in \mathcal{K}$, sample an embedding α_i from a bounded diameter embedding of $K_i^+ = K_i \cup \{ r(K_i) \}$.
4. Use a perfect merge function to sequentially merge each α_i with α_S.

Algorithm 1 formalizes this sampling process, and Theorem 19 states that the embedding sampled by Algorithm 1 samples from a bi-Lipschitz extension of the input embedding $\mathcal{D}_S$. We highlight in red the places where Algorithm 1 uses each of the three components we have discussed so far.

Algorithm 1 Bi-Lipschitz extension: Algorithm for finding a probabilistic bi-Lipschitz extension when the requirements of Theorem 19 are met.

Input: Metric space (X, d), subset $S \subseteq X$, embedding $\mathcal{D}_S : S \to \mathcal{Y}$; the spaces X and $\mathcal{Y}$ should meet the conditions of Theorem 19, and `PerfectMerge` will be the designated perfect merge function

Output: A randomly sampled embedding $\alpha : X \to \mathcal{Y}$.

Sample a partition $(\mathcal{K} = \{ K_i \}_{i \in \mathbb{N}}, r)$ from an onion partition of $K = X \setminus S$
Let K_i^+ be $K_i \cup \{ r(K_i) \}$ for all $K_i \in \mathcal{K}$
Sample $\alpha \sim \mathcal{D}_S$
for each non-empty K_i **do**
    Let $\mathcal{D}_i$ be an expanding bounded-diameter embedding of K_i^+ into $\mathcal{Y}$
    Sample $\alpha_i \sim \mathcal{D}_i$
    $\alpha \leftarrow$ `PerfectMerge`(α, α_i)
Return α

▶ **Theorem 19.** *Let (X, d) be a metric with $S \subseteq X$ and let $\mathcal{D}_S : S \to \mathcal{Y}$ be a probabilistic embedding with contraction at most c'_S and expansion at most c_S.*
Then, if:

1. *X admits an onion partition with separation parameter ρ, closeness 2,[5] and band thickness γ,*
2. *$\mathcal{Y}$ admits expanding bounded-diameter embeddings on sets of size k with expansion c_k and diameter expansion c_Δ, and*
3. *$\mathcal{Y}$ admits a perfect merge function with contraction factor η,*

$\mathcal{D}_S$ has a bi-Lipschitz extension to all of X with extension factors $L_{exp} = O(\gamma \cdot \rho \cdot (c_\Delta / c_S + 1) + c_k / c_S)$ and $L_{contr} = \eta^2$. Further, Algorithm 1 samples from such an extension.

Before proving this theorem, we observe that due to the first two properties of a perfect merge function, it is sufficient to bound the first extension Algorithm 1 produces that includes both x and y in the domain.

▶ **Observation 20.** *Let $x, y \in X$. Consider the first iteration of Algorithm 1 in which both x and y are in the domain of α, and let $\hat{d}$ be $d_\alpha(x, y)$ at that time step. Then at every subsequent iteration of the algorithm, $d_\alpha(x, y) = \hat{d}$.*

We are now ready to bound the contraction and expansion of the embedding procedure, which we do in Lemmas 21 and 22. We defer the full proofs of these lemmas to the full version of this paper and just give a brief overview of the ideas here.

▶ **Lemma 21** (Contraction bound). *Let $X, S, \mathcal{D}_S$ be as in Theorem 19 and let $\alpha : X \to \mathcal{Y}$ be the embedding returned by Algorithm 1. Then for all $x, y \in X$, $d_\alpha(x, y) \geq d(x, y)/\eta^2$.*

[5] This could be defined for more general closeness parameters, but it greatly complicates the algebra, so we set $b = 2$ in our analysis.

To intuitively understand the contraction bound, note that when some node $v \in K_i$ is added to the domain of α, there is no contraction between v and other nodes in K_i^+, as $\mathcal{D}_i$ is non-contracting and we use a perfect merge function to combine this with α. Further, for any node u already in the domain of α when v was added, property 3 of perfect merge functions ensures $d_\alpha(u,v) \geq (d_\alpha(u,r(v)) + d_\alpha(v,r(v)))/\eta$. Applying the triangle inequality and appropriate contraction bounds on α prior to this point gives us the result.

We are now ready to bound the expected expansion for each pair of nodes. This argument follows a similar line of reasoning to that for bi-Lipschitz extensions for ℓ_1 in [26], but because we are using onion partitions rather than CKR partitions, we are able to better bound the distortion in the "bad case" of the embedding.

More specifically, the bad case for the embedding is when a pair of nodes $x, y \in K$ is separated by the partition and they are much closer to each other than they are to S (so routing their path through S potentially causes a large distortion). In this situation, we bound the distance between x and y in terms of $d(x,S)$ and $d(y,S)$, and we use the fact that the probability of separation is bounded by $\rho \cdot \frac{d(x,y)}{d(\{x,y\},S)}$ to obtain our final bound. Crucially, the bound we obtain for the distance between $x \in K_i$ and $y \in K_j$ depends on walking from x to $r(x)$ (the anchor point of x) through the embedding on K_i^+, then from $r(x)$ to $r(y)$ (the anchor point of y) through the embedding on S, and then from $r(y)$ to y through the embedding on K_j^+ (see Figure 2). In the simplest sense, we could apply a bound of c_S for the expansion of the distance between $r(x)$ and $r(y)$ and a bound of c_k (the distortion bound for embedding subsets of size k) for the distances between x and s and between y and t. Because we must pay ρ as part of the separation probability in the partition, our overall expected distortion goes up to $O(\rho \cdot (c_S + c_k))$.[6]

However, because of the onion partitions we use here, $d(x,s)$ is forced to be within a factor of γ of the diameter Δ_x of the cluster x is in. Further, because we use diameter-bounded embeddings on the set K_i^+, we know that the maximum distance between any pair of nodes in that embedding is at most $c_\Delta \cdot \Delta_x$. Thus, the distance between x and $r(x)$ in this embedding is within a constant factor of their true distance, allowing us to reduce our expansion factor in this bad case to $O(\rho \cdot \gamma \cdot (c_S + c_\Delta))$. We potentially still incur a cost of c_k if x and y are in the same component, so our overall cost is $O((c_S + c_\Delta) \cdot \rho \cdot \gamma + c_k)$, which gives us a potential savings if c_S and c_Δ are much smaller than c_k and γ is a constant. Our analysis follows a similar reasoning to that of Calinescu, Karloff, and Rabani [22] for the 0-extension problem, though we deal with an additional merge step and resulting effect on the embedding. See the full version of the paper for a full proof.

▶ **Lemma 22** (Expansion bound). *Let $X, S, \mathcal{D}_S$ be as in Theorem 19, and let $\mathcal{D}$ be the distribution over embeddings from which Algorithm 1 draws when given inputs $X, S, \mathcal{D}_S$. Then for all $x, y \in X$*

$$\mathop{\mathbb{E}}_{\alpha \sim \mathcal{D}}[d_\alpha(x,y)] \leq O(c_k + \gamma \cdot \rho \cdot (c_S + c_\Delta)) \cdot d(x,y),$$

where c_k is the bound on the expansion of the bounded-diameter embeddings for each part of the partition used by the algorithm, c_S is the expansion of $\mathcal{D}_S$, c_Δ is the diameter expansion of the bounded-diameter embeddings used by the algorithm, and $k = |X \setminus S|$.

[6] In fact, this bound applies even if we do not have a bounded diameter embedding (i.e. we just have an embedding with expected expansion c_k on sets of size k) and even if the partitions we use do not meet the third criterion of onion partitions (requiring that the distance between a cluster node and S be similar to the diameter of the cluster).

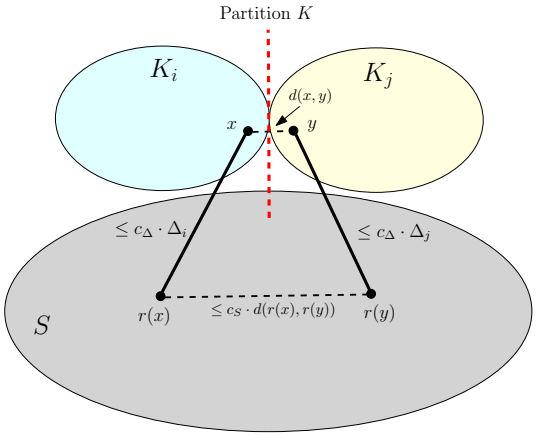


Figure 2 The "bad case" routing scenario for expansion bounds. Nodes $x \in K_i$ and $y \in K_j$ are physically close but separated by the onion terminal partition $\mathcal{K}$. The expected distance between x and y in the merged embedding is bounded by routing the path through their respective anchor points $r(x)$ and $r(y)$ within the base embedding S.

Noting that Algorithm 1 always outputs an embedding in which the distances on pairs of points in S are the same as under the sampled embedding α_S, Theorem 2 (finding bi-Lipschitz extensions for HSTs) is immediate from Theorems 12, 14, and 19 and Lemma 17. Theorem 3 (finding bi-Lipschitz extensions for ℓ_1) takes a little more work, and we prove it in Section 5.

4 Onion partitions

In this section we prove Theorem 12.

▶ **Theorem 12** (Onion partitions). *For any metric (X, d) and any $S \subseteq X$, there exists an onion partition on $K = X \setminus S$ with separation parameter $O(\log k)$, closeness 2, and band thickness 6, where $k = |K|$.*

We begin with a discussion of the partition algorithm given by Calinescu, Karloff, and Rabani in their work on the 0-extension problem [22], which we refer to as CKR partitions. Our onion partitions will consist of CKR partitions intersected with a randomized bucketing scheme for $K = X \setminus S$. The CKR partitions ensure that nodes are separated with a probability proportional to their distance, and the randomized bucketing allows us to control the diameter of each cluster relative to the designated anchor point without increasing the probability of separation between x and y by too much.

Given a metric (X, d) and a subset $S \subseteq X$, [22] finds a partition $\{K_1, K_2, \ldots, K_{|S|}\}$ of $K := X \setminus S$ such that the "anchor point" of K_s is $s \in S$, the probability of $x, y \in K$ being separated is $O(\log k \cdot \frac{d(x,y)}{d(\{x,y\},S)})$, and $x \in K_s$ for $s \in S$ implies $d(x, s) \leq 2d(x, S)$. Algorithm 2 presents a variation of the CKR algorithm in which we use a subset $T \subseteq S$ as the set of anchor points, rather than the entire set S.

The CKR partitioning algorithm works by selecting a random permutation on some set of terminals and then "assigning" points in $X \setminus S$ to terminals in $T \subseteq S$ such that (1) a node in $X \setminus S$ is always assigned to a terminal that is fairly close to it, and (2) the probability that a pair of nodes $x, y \in X$ are placed in separate part of the partition is proportional to $d(x, y)/d(\{x, y\}, T)$. We state these two properties as Observation 23 and Lemma 24. Note that if we let the anchor point of a cluster K_t for $t \in T$ be t itself, then these two facts imply that CKR partitions meet criteria (1) and (2) of onion partitions, so we're left with fulfilling criterion (3).

Algorithm 2 CKR partitioning.

Input: Metric space (X, d), subset $S \subseteq X$, a set of terminals $T \subseteq S$
Output: A partition $\{ K_t \}_{t \in T}$ of $X \setminus S$

1: Sample a permutation π uniformly at random over T
2: Sample $\mu \in [1, 2)$ uniformly at random.
3: **for** Each node $x \in X \setminus S$ **do**
4: Let $t \in T$ be the first terminal under the ordering π such that $d(x, t) \leq \mu \cdot d(x, T)$
5: Add x to K_t
6: Return $\{ K_t \}_{t \in T}$, where the anchor point of K_t is set to t

▶ **Observation 23** (Deterministic Proximity of CKR Rounding, implicit in [22]). *For any $x \in X \setminus S$, if Algorithm 2 assigns x to K_t for some $t \in T$, then*

$$d(x, t) \leq \mu \cdot d(x, T) \leq 2 \cdot d(x, T),$$

where the last inequality holds because $\mu \in [1, 2)$.

▶ **Lemma 24** (CKR Separation, Lemma 2.2 in [22]). *Fix $x, y \in X$ and let $\delta := d(x, y)$. If $0 < \delta \leq \frac{1}{4} \min\{d(x, T), d(y, T)\}$, then:*

$$\Pr[x, y \text{ are separated}] \leq 4\mathcal{H}_{|T|} \left(\frac{\delta}{d(x, T)} + \frac{\delta}{d(y, T)} \right),$$

where $\mathcal{H}_m = 1 + \frac{1}{2} + \cdots + \frac{1}{m}$ denotes the m-th harmonic number.

For our partitions, we will pick T to be the set of "nearest neighbors" of nodes in K. That is, $T := \{ t \in \arg\min_{s \in S} \{ d(s, x) \} \mid x \in K \}$. We assume that for each $x \in K$, we add only one of the nearest neighbors of x to T so that $|T| \leq k$. Moreover, by the choice of T, we also have $d(x, T) = d(x, S)$ for all $x \in K$.

Randomized bucketing

Now we show how to refine the CKR partition technique so that the diameter of each partition is bounded by a constant factor times the distance from that partition to the terminal set T. In particular, we will use a technique called *randomized scale partitioning*. For any point $x \in K$, let $A_x = d(x, T)$ denote its distance to the nearest terminal. We choose a shift parameter $u \in [0, 1)$ uniformly at random and define $\beta = 2^u$. We then (randomly) bucket K into disjoint scales B_i by distance from T:

$$B_i := \{x \in K \mid \beta \cdot 2^i \leq A_x < \beta \cdot 2^{i+1}\}.$$

Given a metric (X, d) with diameter Δ, subset $S \subseteq X$, and terminal set $T \subseteq S$, we will randomly partition $K = X \setminus S$ into sets $\{ K_t \}_{t \in T}$ and buckets $\{ B_i \}_{i \in [\lceil \log \Delta \rceil]}$,[7] where the K_t are chosen by finding a CKR partition on K with terminal set T and the B_i are chosen by randomized scale partition. Then we define a new partition $\{ K_{t,i} \}_{t \in T, i \in [\lceil \log \Delta \rceil]}$, where $K_{t,i} := (K_t \cap B_i)$ and let the anchor point of this set be t – that is, $r(K_{t,i}) = t$.

[7] Here, $[n]$ denotes $\{ 0, 1, 2, \ldots, n \}$.

The primary properties of this partitioning that we will use are (1) for any node $x \in K_{t,i}$, the distance from x to another node in $K_{t,i}$ is not too much larger than the distance from x to T, and (2) the probability that a pair of nodes x and y are separated in this partition is not too much larger than the probability that x and y are separated under the CKR embedding. We express these as Observation 25 and Lemma 26, respectively.

▶ **Observation 25** (Partition diameter bound). *For any part $K_{t,i}$ of our scale partition and any node $x \in K_{t,i}$, if $\Delta_{t,i}$ is the diameter of $K_{t,i}$, then $d(x,t) \geq \Delta_{t,i}/6$*

Proof. For any pair $x, y \in K_{t,i}$, we have $d(x,y) \leq d(x,t) + d(t,y) \leq 2d(x,T) + 2d(y,T) \leq 6d(x,T)$. The first inequality is by the deterministic properties of CKR partitions, and the second is because x and y being in the same bucket implies $d(y,T)$ is no more than twice $d(x,T)$. Thus, $\Delta_{t,i} \leq 6 \cdot d(x,T) \leq 6 \cdot d(x,t)$. ◀

▶ **Lemma 26** (Scale Partition Separation). *For any pair of points $x, y \in K = X \setminus S$, let $\delta := d(x,y)$, and assume without loss of generality that $A_x \leq A_y$. The probability that x and y assigned to distinct buckets B_i and B_j is bounded by $O\left(\frac{\delta}{A_x}\right)$.*

Thus, by the union bound and Lemma 24, the probability that x and y are in different parts of the partition $\{ K_{t,i} \}$ is at most $O(\log |T| \cdot \frac{\delta}{A_x})$.

Proof. Let x be in bucket i and y be in bucket j. The scale boundaries are determined by $\beta 2^i = 2^{i+u}$, which is equivalent to placing a grid of integers shifted by a uniformly random $u \in [0,1)$ on a logarithmic scale. The probability that an integer boundary falls strictly between $\log_2 A_x$ and $\log_2 A_y$ is bounded by the length of the interval between their logarithmic distances. By the metric triangle inequality, $A_y \leq A_x + \delta$. Therefore:

$$\begin{aligned}
\Pr[i \neq j] &\leq \log_2(A_y) - \log_2(A_x) \\
&= \log_2\left(\frac{A_y}{A_x}\right) \\
&\leq \log_2\left(\frac{A_x + \delta}{A_x}\right) = \log_2\left(1 + \frac{\delta}{A_x}\right) \\
&\leq \frac{\delta}{A_x \ln 2} \;=\; O\left(\frac{\delta}{A_x}\right)
\end{aligned}$$

where the final inequality follows from the standard logarithmic bound $\ln(1+z) \leq z$ for all $z \geq 0$. ◀

Thus, the partitions we have constructed inherit property (2) of onion partitions with closeness 2 from the CKR bounds on anchor point distances and our choice of T (as the nearest neighbors of nodes in K), and they inherit property (3) of onion partitions with band thickness 6 from bounds on the diameter of randomized bucketing (and the fact that taking a subset of a cluster never increases its diameter). They also achieve property (1) with parameter $O(\log k)$ by Lemmas 24 and 26, our choice of T, and a union bound. Thus, we get Theorem 12.

5 Merge functions for HSTs and ℓ_1

In this section, we give explicit perfect merge functions for ℓ_1 and HSTs, and we prove Theorem 3.

Algorithm 3 `MergeL1`(α_1, α_2) Algorithm for merging two ℓ_1 embeddings.

Input: Embeddings $\alpha_1 : Z_1 \to \ell_1$, $\alpha_2 : Z_2 \to \ell_1$, where $|Z_1 \cap Z_2| = 1$.
Output: An embedding of $Z_1 \cup Z_2$ into ℓ_1

1: Let $u = Z_1 \cap Z_2$
2: For all $x \in Z_1$, let $\alpha(x) \leftarrow \alpha_1(x) \circ \alpha_2(u)$, where $\circ$ is concatenation
3: For all $x \in Z_2$, let $\alpha(x) \leftarrow \alpha_1(u) \circ \alpha_2(x)$
4: Return α

5.1 Merge for ℓ_1

We begin by presenting a perfect merge function for ℓ_1 embeddings. This is captured in Algorithm 3, which essentially just extends each input embedding to the entire set $Z_1 \cup Z_2$ by mapping everything not already in the domain to the same point as $u \in Z_1 \cap Z_2$ and concatenating the resulting embeddings together. This is easily seen to meet the requirements of a perfect merge function with contraction bound 1 and thus we get Lemma 18 which states that ℓ_1 has perfect merge functions. We now move on to proving Theorem 3.

▶ **Theorem 3** (ℓ_1 has bi-Lipschitz extensions). *For any subset $S \subseteq X$ with an embedding $\alpha_S : S \xhookrightarrow{c_S, c'_S} \ell_1$, there exists*

1. *a deterministic* weak *bi-Lipschitz extension of α_S, $\alpha : X \hookrightarrow \ell_1$ with extension factors $L_{exp} = O(\log k)$ and $L_{contr} = 1$, where $k := |X \setminus S|$ and*
2. *a deterministic strong bi-Lipschitz extension with $L_{exp} = O(\log k)$ and $L_{contr} = 7$.*

Proof. Part (1) of the proof is almost immediate from Theorems 12, 14, and 19, the fact that ℓ_1 has a perfect merge function with contraction factor 1, and the fact that Algorithm 1 outputs an embedding in which pairs of nodes in S have the same distance as under α_S. The problem is that Theorem 19 implies the existence of a *probabilistic* extension of the original embedding. However, for any finite support probabilistic embedding $\mathcal{D}$ of X into ℓ_1, there exists a deterministic embedding $\alpha : X \to \ell_1$ with $d_\alpha(x, y) = \mathbb{E}_{\alpha' \sim \mathcal{D}}[d_{\alpha'}(x, y)]$. (Note that the dimension of α may be much larger than those of the embeddings in the support of $\mathcal{D}$, and it is not clear how to efficiently find α when $\mathcal{D}$ has a large support.) See e.g. [26] for more. Proving the second part of this theorem requires a bit more of a sophisticated de-randomization method, and we defer it to the full version of the paper. ◀

5.2 Merge for HSTs

We now present a perfect merge function for HSTs to prove Lemma 17, which states that HSTs have perfect merge functions with contraction factor 2.

Algorithm 4 takes in two β-HSTs for a fixed ratio β and returns another β-HST. This algorithm works by "combining" the ancestors of the common point u at each level of the two HSTs. That is, the new HST consists of u and all its ancestors, and each ancestor of u has as its child subtrees the child subtrees of the ancestor of u at its same level in each HST. Figure 3 gives a visual depiction of this algorithm. We defer the correctness proof for Lemma 17 to the full version of the paper and briefly outline the idea here.

For any $x, y \in Z_1$ or $x, y \in Z_2$, the least common ancestor of x and y must be at the same level in the new tree as it was in the original embedding of Z_1 or Z_2, so their distance is unchanged from the original embedding. To bound the distance between $x \in Z_1, y \in Z_2$, consider the least common ancestor of x and y in the HST produced at the algorithm. The first ancestor of y that has descendants from Z_1 is the least common ancestor of y and u.

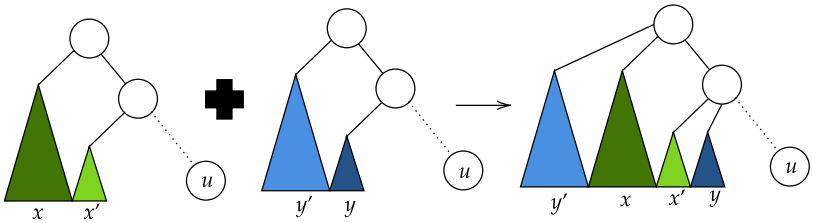


Figure 3 This figure depicts what happens in the algorithm `MergeHST`. In particular, u is the node common to both HSTs on the left side of the figure. The triangles represent subtrees, and the letters below the triangles indicate the presence of the corresponding node as a leaf of that subtree. Note that after merging trees, the least common ancestor of x and y is that of x and u from the first tree, as this was higher than that of y and u. This is the foundational idea behind why $d_\alpha(x,y) \geq \max\{ d_{\alpha_1}(u,x), d_{\alpha_2}(u,y) \}$ holds when α is the output embedding for `MergeHST` on inputs α_1 and α_2.

Thus, the least common ancestor of x and y cannot be lower than the least common ancestor of y and u, so the distance between x and y is at least that between y and u under the original embedding on Z_2. Likewise, the distance between x and y is at least the distance between x and u under the original embedding on Z_1, so we get contraction at most 2.

Algorithm 4 `MergeHST`(α_1, α_2) Algorithm for merging two β-HSTs.

Input: Embeddings $\alpha_1 : Z_1 \to T_1$, $\alpha_2 : Z_2 \to T_2$ for β-HSTs T_1, T_2, where $|Z_1 \cap Z_2| = 1$. We assume without loss of generality that the height of T_1 is at least as large as the height of T_2.
Output: An embedding of $Z_1 \cup Z_2$ into a single β-HST

Let $u = Z_1 \cap Z_2$
Create a new HST T that is initially a copy of T_1
$\alpha(x) \leftarrow \alpha_1(x)$ for all $x \in Z_1$
for $i = 0$ to the height of T_1 **do**
    for the subtree of each child of $\alpha_2(u)$'s ancestor at height i in T_2 **do**
        if the subtree does not contain $\alpha_2(u)$ **then** ▷ i.e. if the subtree is not yet merged into T_1
            Copy the subtree and make it a child of $\alpha_1(u)$'s ancestor at level i in T
            for any $x \in Z_2 \setminus \{ u \}$ such that $\alpha_2(x)$ is in the subtree at this step **do**
                Set $\alpha(x)$ to be the copy of $\alpha_2(x)$ created in T at this step
Return α

6 Outlier embeddings for HSTs

Now, given the existence of bi-Lipschitz extensions for HSTs, we turn to approximating outlier embeddings into HSTs. In [26], this type of approximation is obtained for ℓ_2 by introducing new "outlier variables" into a semidefinite program for a minimum distortion ℓ_2 embedding and rounding. We will employ a similar technique here using the linear program and associated rounding algorithm of [53] to approximate the optimal distortion for probabilistic HST embeddings.

Let Δ be the diameter of (X,d) and let $M = \{ 2^0, 2, 2^2, \ldots, 2^{\lceil \log \Delta \rceil} \}$. Let ζ be the constant factor from Theorem 2 and let $B_i(r) := \{ j \in X : d(i,j) \leq r \}$. We define (HST LP), which is parameterized by inputs $((X,d),k,c)$. The changes from [53] are highlighted in red.

Table 1 We imagine assigning a "representative" at each node of the HST such that the representative of a leaf node is the leaf node itself, and the representative of any other node is the same as one of its children's representatives. This table describes the intuitive meaning of each variable in the LP (which is largely the same as in [53]).

variable	intuitive meaning
δ_i	indicator for whether i is an outlier
$z^r_{ijj'}$	the probability that i is the representative of j and j' at level r
$\gamma^r_{jj'}$	the probability that j and j' are separated at level r
x^r_{ij}	the probability that i is the representative of j at level r

Table 1 gives an intuitive description of the variables in the LP. Note that in our LP c is a parameter, but in [53] it is the variable they are minimizing. The biggest change from [53] is that we have introduced new variables δ_i for each $i \in X$ that we use as an indicator for whether i is an outlier in the embedding.

$$\min \sum_{i=1}^{n} \delta_i \tag{HST LP}$$

$$\forall j, j' \sum_{r \in M} r \cdot \gamma^r_{jj'} \leq (4 + \zeta \cdot (\delta_j + \delta_{j'}) \cdot \log k) \cdot c \cdot d(j, j') \tag{7}$$

$$\forall i, r, (j, j' \in B_i(r)) \quad \min(x^r_{i,j}, x^r_{ij'}) \geq z^r_{ijj'} \tag{8}$$

$$\forall j, j', i, r \sum_{i: j, j' \in B_i(r)} z^r_{ijj'} \geq 1 - \gamma^r_{jj'} \tag{9}$$

$$\forall j, r \sum_{i: j \in B_i(r)} x^r_{ij} = 1 \tag{10}$$

$$\forall j, j', (r < d(j, j')) \quad \gamma^r_{jj'} = 1 \tag{11}$$

$$\forall j, j', r \quad \gamma^r_{jj'} \leq 1 \tag{12}$$

$$\forall i, j, j', r \quad \gamma^r_{jj'} x^r_{ij}, z^r_{ijj'}, \delta_i \geq 0; \delta_i \leq 1 \tag{13}$$

Given an optimal solution to this linear program, we need to define an outlier embedding with low distortion and few outliers. To do so, we will first use the rounding algorithm of [53] to find an embedding on the entire metric, and then we take as outliers all vertices i such that $\delta_i \geq \delta^*$ for some threshold δ^*. In the full version of the paper we formally prove that this rounding algorithm works and gives us Theorem 1 as a result. In a similar manner to ℓ_2 outlier embeddings in [26], we use the existence of nested embeddings to bound the optimal LP solution and thus obtain our final result.

Further, if we modify (HST LP) by changing the objective function to $\sum_i w_i \cdot \delta_i$ for some set of weights w_i, we can use the same technique to obtain a bicriteria approximation for the weighted variant of the outlier embedding problem, where the cost of making i an outlier is w_i. This result is expressed as Corollary 10, and we prove it in the full version of the paper.

7 Applications

In the Minimum Cost Communication Tree (MCCT) problem, we are given a finite metric $\mathcal{X} = (X, d)$ and weights r_{ij} for all pairs of nodes i, j and we are asked for a non-contracting embedding α of X into a tree with vertex set X such that $\sum_{i,j \in X} r_{ij} \cdot d_\alpha(i, j)$ is minimized. This problem is motivated by evolutionary biology, in which a scientist may attempt to

reconstruct a tree of relationships between multiple organisms, and she comes to the problem with some known minimum distances between them, derived through a method such as DNA sequence analysis. In such a setting, the r_{ij} are an additional tunable parameter that can allow the scientist to indicate pairs she considers "more likely" to be related, by giving them a bigger r_{ij} value. As noted by Wu et al. [60], this problem can be approximated in expectation to within a factor of $O(\log n)$ by simply embedding randomly into an HST and converting it to a tree with vertex set X. Directly applying the algorithm of Munagala et al. [53] with the observations of Wu et al. [60], we have a randomized algorithm that gets an $O(c(\mathcal{X}))$-approximation for the MCCT problem, where $c(\mathcal{X})$ is the minimum distortion needed to probabilistically embed $\mathcal{X}$ into HSTs.

It is quite natural to consider an outlier version of the MCCT problem. In the evolutionary biology setting, we could consider the case that we are given some noisy data points, or points that don't really belong in the set being analyzed. One approach to this type of outlier problem may be to look for a k outlier tree embedding of minimum distortion, but the set of outliers that minimizes the embedding distortion does not necessarily also minimize the MCCT solution cost. We may expect that the outlier MCCT problem is easier to solve on tree metrics than on general metrics, so we could instead try the following approach: embed the input metric into a tree, and *then* try to find the best outliers for the resulting tree metric. In fact, if no outliers are allowed, MCCT is trivial on tree metrics. Further, in Subsection 7.1 we give a bicriteria approximation to the k-outlier MCCT problem when the input is a tree metric. In particular, if the optimal cost of a k outlier solution for a given instance of a tree metric is OPT_k, then we give an algorithm that finds a solution with at most $3k$ outliers and at most $24OPT_k$ cost. Let $c_k(\mathcal{X})$ be the minimum distortion required for probabilistically embedding all but at most k points in $\mathcal{X}$ into HSTs. We can use the outlier embedding algorithm from this paper to embed any input metric into a tree with outliers and then apply the approximation algorithm for outlier MCCT on trees, obtaining a solution with at most $O(\frac{k}{\epsilon}\log k)$ outliers and expected cost $O((1+\epsilon)\cdot c_k(\mathcal{X})\cdot OPT_k)$. By contrast, directly using the embedding algorithm of [53] with the tree approximation algorithm from this section instead gives a solution with at most $3k$ outliers and $O((1+\epsilon)\cdot c_0(\mathcal{X})\cdot OPT_k)$ expected cost, so we are unable to save on the cost of the optimal solution if there is a large subset that embeds well into HSTs but the entire metric does not. We explain this further in the full version of the paper.

7.1 MCCT with outliers

Let (T,d) be a tree metric, and let k be an integer. Then we define variables x_{ij} for each pair of nodes ij and variables δ_i for each variable i. We can use them to make the following linear program.

$$\min \sum_{i,j\in T} x_{ij}\cdot r_{ij}\cdot d(i,j) \quad s.t. \qquad \text{(MCCT Outlier LP)}$$

$$\forall i,j\in T: \quad \delta_i+\delta_j+x_{ij}\geq 1$$

$$\sum_{i\in T}\delta_i \leq k$$

$$\forall i,j\in T: \quad \delta_i, x_{ij}\in[0,1]$$

▷ Claim 27. Let (T,d) be a tree metric such that there exists a subset $K\subseteq T$, $|K|\leq k$, and $\sum_{i,j\in T\setminus K} r_{ij}\cdot d(i,j)\leq D$.

Then the following procedure finds a subset $K' \subseteq T$ with $|K'| \leq 3k$ and an embedding α of $T \setminus K'$ into a tree with the same vertex set and $\sum_{i,j \in T \setminus K'} d_\alpha(i,j) \leq 24D$.

1. Solve MCCT Outlier LP and let x_{ij}, δ_j be the resulting values
2. Let $K' = \{ i : \delta_i \geq 1/3 \}$
3. Embed $T \setminus K'$ into a tree T' with distortion at most 8, using the algorithm of Gupta [43]

Proof. First, MCCT Outlier LP has a solution of value at most D. In particular, let K be as in the claim statement and consider setting $\delta_i = \begin{cases} 1 & i \in K \\ 0 & else \end{cases}$ and $x_{ij} = \begin{cases} 1 & \delta_i = \delta_j = 0 \\ 0 & else \end{cases}$. As there are only k outliers, the second constraint is clearly satisfied, and we have defined x_{ij} such that the other constraints are also satisfied. Further, x_{ij} is 0 if either i or j is an outlier in K, so the objective function value is just $\sum_{i,j \in T \setminus K} r_{ij} d(i,j)$, which by the claim statement is bounded by D.

Let K' be as in the claim. As the δ_i sum to at most k, we know $|K'| \leq 3k$. Consider any x_{ij} with $i, j \notin K'$ We have $x_{ij} \geq 1 - \delta_i - \delta_j \geq 1/3$ by how we chose K'. The cost associated with a given pair ij of non-outliers in our solution is $r_{ij} d(i,j)$, but that same pair contributed at least $r_{ij} d(i,j)/3$ to the optimal objective function value, so our final cost is at most 3 times the objective, which is at most $3D$. At this point, we have an isometric (distortion 1) embedding of the metric induced by $T \setminus K'$ into the original metric T itself. Thus, we can view K' as a set of Steiner points in our original metric.

Finally we consider the loss incurred applying the algorithm of Gupta [43]. This creates a tree embedding that removes Steiner points and incurs distortion at most 8, so the cost of this embedding on $T \setminus K'$ is at most $24D$, as desired. This step is what prevents us from applying this algorithm directly to general metrics. ◁

8 Open questions

There are still many open questions we can ask about topics in this space, such as:

- Is it NP-hard to find the optimal outlier set size when probabilistically embedding a given metric into HSTs with a small fixed distortion? This question is known to be hard for deterministic embeddings into some spaces [59, 26].
- Even if finding the optimal outlier set size for probabilistically embedding into HSTs is NP-hard, can the approximation factors in this paper be improved?
- Which other metric spaces admit perfect merge functions (and thus bi-Lipschitz extensions), and can we approximate optimal outlier embeddings in such spaces?

References

1 I. Abraham, Y. Bartal, J. Kleinberg, T.-H.H. Chan, O. Neiman, Kedar Dhamdhere, A. Slivkins, and Anupam Gupta. Metric embeddings with relaxed guarantees. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, pages 83–100, 2005. doi:10.1109/SFCS.2005.51.

2 Ittai Abraham, Yair Bartal, and Ofer Neiman. On low dimensional local embeddings. In *Proceedings of the 2009 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 875–884, 2009. doi:10.1137/1.9781611973068.95.

3 Ittai Abraham, Yair Bartal, and Ofer Neiman. Advances in metric embedding theory. *Advances in Mathematics*, 228(6):3026–3126, 2011. doi:10.1016/j.aim.2011.08.003.

4 Ittai Abraham, Yair Bartal, and Ofer Neiman. Embedding metrics into ultrametrics and graphs into spanning trees with constant average distortion. *SIAM J. Comput.*, 44(1):160–192, 2015. doi:10.1137/120884390.

5 Ittai Abraham, Shiri Chechik, Michael Elkin, Arnold Filtser, and Ofer Neiman. Ramsey spanning trees and their applications. *ACM Trans. Algorithms*, 16(2):19:1–19:21, 2020. preliminary version published in SODA 2018. doi:10.1145/3371039.

6 Noga Alon, Mihai Badoiu, Erik D. Demaine, Martin Farach-Colton, Mohammad Taghi Hajiaghayi, and Anastasios Sidiropoulos. Ordinal embeddings of minimum relaxation: General properties, trees, and ultrametrics. *ACM Trans. Algorithms*, 4(4):46:1–46:21, 2008. doi:10.1145/1383369.1383377.

7 Noga Alon, Richard M Karp, David Peleg, and Douglas West. A graph-theoretic game and its application to the k-server problem. *SIAM Journal on Computing*, 24(1):78–100, 1995. doi:10.1137/S0097539792224474.

8 Sanjeev Arora, László Lovász, Ilan Newman, Yuval Rabani, Yuri Rabinovich, and Santosh Vempala. Local versus global properties of metric spaces. *SIAM Journal on Computing*, 41(1):250–271, 2012. doi:10.1137/090780304.

9 Mihai Badoiu, Julia Chuzhoy, Piotr Indyk, and Anastasios Sidiropoulos. Low-distortion embeddings of general metrics into the line. In Harold N. Gabow and Ronald Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 225–233. ACM, 2005. doi:10.1145/1060590.1060624.

10 Mihai Badoiu, Julia Chuzhoy, Piotr Indyk, and Anastasios Sidiropoulos. Embedding ultrametrics into low-dimensional spaces. In Nina Amenta and Otfried Cheong, editors, *Proceedings of the 22nd ACM Symposium on Computational Geometry, Sedona, Arizona, USA, June 5-7, 2006*, pages 187–196. ACM, 2006. doi:10.1145/1137856.1137886.

11 Mihai Badoiu, Piotr Indyk, and Anastasios Sidiropoulos. Approximation algorithms for embedding general metrics into trees. In Nikhil Bansal, Kirk Pruhs, and Clifford Stein, editors, *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pages 512–521. SIAM, 2007. URL: http://dl.acm.org/citation.cfm?id=1283383.1283438.

12 Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 184–193, 1996. doi:10.1109/SFCS.1996.548477.

13 Yair Bartal. On approximating arbitrary metrices by tree metrics. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 161–168, New York, NY, USA, 1998. Association for Computing Machinery. doi:10.1145/276698.276725.

14 Yair Bartal. Graph decomposition lemmas and their role in metric embedding methods. In *Algorithms - ESA 2004, 12th Annual European Symposium, Bergen, Norway, September 14-17, 2004, Proceedings*, pages 89–97, 2004. doi:10.1007/978-3-540-30140-0_10.

15 Yair Bartal. Advances in metric ramsey theory and its applications. *CoRR*, abs/2104.03484, 2021. arXiv:2104.03484.

16 Yair Bartal, Arnold Filtser, and Ofer Neiman. On notions of distortion and an almost minimum spanning tree with constant average distortion. *J. Comput. Syst. Sci.*, 105:116–129, 2019. preliminary version published in SODA 2016. doi:10.1016/j.jcss.2019.04.006.

17 Yair Bartal, Nathan Linial, Manor Mendel, and Assaf Naor. On metric ramsey-type phenomena. In *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*, STOC '03, pages 463–472, New York, NY, USA, 2003. Association for Computing Machinery. doi:10.1145/780542.780610.

18 Yair Bartal, Nathan Linial, Manor Mendel, and Assaf Naor. On metric Ramsey-type dichotomies. *Journal of the London Mathematical Society*, 71(2):289–303, 2005. doi:10.1112/S0024610704006155.

19 Yair Bartal, Nathan Linial, Manor Mendel, and Assaf Naor. Some low distortion metric ramsey problems. *Discret. Comput. Geom.*, 33(1):27–41, 2005. doi:10.1007/s00454-004-1100-z.

20 Guy E. Blelloch, Yan Gu, and Yihan Sun. A new efficient construction on probabilistic tree embeddings. *CoRR*, abs/1605.04651, 2016. arXiv:1605.04651.

21 J. Bourgain, T. Figiel, and V. Milman. On Hilbertian subsets of finite metric spaces. *Israel J. Math.*, 55(2):147–152, 1986. doi:10.1007/BF02801990.

22 Gruia Calinescu, Howard Karloff, and Yuval Rabani. Approximation algorithms for the 0-extension problem. *SIAM Journal on Computing*, 34(2):358–372, 2005. doi:10.1137/S0097539701395978.

23 T. H. Hubert Chan, Michael Dinitz, and Anupam Gupta. Spanners with slack. In Yossi Azar and Thomas Erlebach, editors, *Algorithms – ESA 2006*, pages 196–207, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. doi:10.1007/11841036_20.

24 Moses Charikar, Chandra Chekuri, Ashish Goel, Sudipto Guha, and Serge Plotkin. Approximating a finite metric by a small number of tree metrics. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, FOCS '98, page 379, USA, 1998. IEEE Computer Society.

25 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Local global tradeoffs in metric embeddings. *SIAM Journal on Computing*, 39(6):2487–2512, 2010. doi:10.1137/070712080.

26 Shuchi Chawla and Kristin Sheridan. Composition of nested embeddings with an application to outlier removal. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1641–1668, 2024. doi:10.1137/1.9781611977912.66.

27 Victor Chepoi, Feodor F. Dragan, Ilan Newman, Yuri Rabinovich, and Yann Vaxès. Constant approximation algorithms for embedding graph metrics into trees and outerplanar graphs. *Discret. Comput. Geom.*, 47(1):187–214, 2012. doi:10.1007/s00454-011-9386-0.

28 Yeshwanth Cherapanamjeri and Jelani Nelson. Terminal embeddings in sublinear time. *TheoretiCS*, 3, 2024. doi:10.46298/THEORETICS.24.6.

29 Karine Chubarian and Anastasios Sidiropoulos. Computing bi-lipschitz outlier embeddings into the line. In *International Workshop and International Workshop on Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 36:1–36:21, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.36.

30 Mark de Berg, Krzysztof Onak, and Anastasios Sidiropoulos. Fat polygonal partitions with applications to visualization and embeddings. *J. Comput. Geom.*, 4:212–239, 2010.

31 Jeff Edmonds, Anastasios Sidiropoulos, and Anastasios Zouzias. Inapproximability for planar embedding problems. In Moses Charikar, editor, *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 222–235. SIAM, 2010. doi:10.1137/1.9781611973075.20.

32 M. Elkin, A. Filtser, and O. Neiman. Terminal embeddings. *Theoretical Computer Science*, 697:1–36, 2017. doi:10.1016/j.tcs.2017.06.021.

33 Michael Elkin, Arnold Filtser, and Ofer Neiman. Prioritized metric structures and embedding. *SIAM J. Comput.*, 47(3):829–858, 2018. doi:10.1137/17M1118749.

34 Michael Elkin and Ofer Neiman. Lossless prioritized embeddings. *SIAM J. Discret. Math.*, 36(3):1529–1550, 2022. doi:10.1137/21M1436221.

35 Matthias Englert, Anupam Gupta, Robert Krauthgamer, Harald Räcke, Inbal Talgam-Cohen, and Kunal Talwar. Vertex sparsifiers: New results from old techniques. *SIAM J. Comput.*, 43(4):1239–1262, 2014. doi:10.1137/130908440.

36 Jittat Fakcharoenphol, Chris Harrelson, Satish Rao, and Kunal Talwar. An improved approximation algorithm for the 0-extension problem. In *SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 257–265, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=644108.644153.

37 Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 448–455, 2003. doi:10.1145/780542.780608.

38 Michael R. Fellows, Fedor V. Fomin, Daniel Lokshtanov, Elena Losievskaja, Frances A. Rosamond, and Saket Saurabh. Distortion is fixed parameter tractable. *ACM Trans. Comput. Theory*, 5(4):16:1–16:20, 2013. doi:10.1145/2489789.

39 Arnold Filtser, Yuval Gitlitz, and Ofer Neiman. Light, reliable spanners. In Wolfgang Mulzer and Jeff M. Phillips, editors, *40th International Symposium on Computational Geometry, SoCG 2024, June 11-14, 2024, Athens, Greece*, volume 293 of *LIPIcs*, pages 56:1–56:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.SOCG.2024.56.

40 Arnold Filtser, Robert Krauthgamer, and Ohad Trabelsi. Relaxed voronoi: A simple framework for terminal-clustering problems. In Jeremy T. Fineman and Michael Mitzenmacher, editors, *2nd Symposium on Simplicity in Algorithms, SOSA 2019, January 8-9, 2019, San Diego, CA, USA*, volume 69 of *OASIcs*, pages 10:1–10:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/OASIcs.SOSA.2019.10.

41 Arnold Filtser and Hung Le. Clan embeddings into trees, and low treewidth graphs. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 342–355. ACM, 2021. doi:10.1145/3406325.3451043.

42 Arnold Filtser and Hung Le. Low treewidth embeddings of planar and minor-free metrics. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 1081–1092. IEEE, 2022. doi:10.1109/FOCS54457.2022.00105.

43 Anupam Gupta. Steiner points in tree metrics don't (really) help. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 220–227, 2001. URL: http://dl.acm.org/citation.cfm?id=365411.365448.

44 William B. Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into hilbert space. *Contemporary mathematics*, 26:189–206, 1984. URL: https://api.semanticscholar.org/CorpusID:117819162.

45 Jon Kleinberg, Aleksandrs Slivkins, and Tom Wexler. Triangulation and embedding using small sets of beacons. *J. ACM*, 56(6):32:1–32:37, September 2009. doi:10.1145/1568318.1568322.

46 Rasmus Kyng, Maximilian Probst Gutenberg, and Tim Rieder. Random-shift revisited: Tight approximations for tree embeddings and ℓ_1-oblivious routings. In *66th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2025, Sydney, Australia, December 14-17, 2025*, pages 2132–2142. IEEE, 2025. doi:10.1109/FOCS63196.2025.00113.

47 Christiane Lammersen, Anastasios Sidiropoulos, and Christian Sohler. Streaming embeddings with slack. In Frank Dehne, Marina Gavrilova, Jörg-Rüdiger Sack, and Csaba D. Tóth, editors, *Algorithms and Data Structures*, pages 483–494, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. doi:10.1007/978-3-642-03367-4_42.

48 J. R. Lee and A. Naor. Extending lipschitz functions via random metric partitions. *Inventiones Mathematicae*, 160(1):59–95, 2005. doi:10.1007/s00222-004-0400-5.

49 Sepideh Mahabadi, Konstantin Makarychev, Yury Makarychev, and Ilya P. Razenshteyn. Nonlinear dimension reduction via outer bi-lipschitz extensions. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 1088–1101. ACM, 2018. doi:10.1145/3188745.3188828.

50 Jiří Matoušek. On embedding expanders into ℓ_p spaces. *Israel Journal of Mathematics*, 102(1):189–197, 1997.

51 Jiří Matoušek and Anastasios Sidiropoulos. Inapproximability for metric embeddings into R^d. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 405–413. IEEE Computer Society, 2008. doi:10.1109/FOCS.2008.21.

52 Manor Mendel and Assaf Naor. Ramsey partitions and proximity data structures. *Journal of the European Mathematical Society*, 9(2):253–275, 2007.

53 Kamesh Munagala, Govind S. Sankar, and Erin Taylor. Probabilistic Metric Embedding via Metric Labeling. In Nicole Megow and Adam Smith, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2023)*, volume 275 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:10, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX/RANDOM.2023.2.

54 Assaf Naor and Yuval Rabani. On lipschitz extension from finite subsets. *Israel Journal of Mathematics*, 219, June 2015. doi:10.1007/s11856-017-1475-1.

55 Assaf Naor and Terence Tao. Scale-oblivious metric fragmentation and the nonlinear dvoretzky theorem. *Israel Journal of Mathematics*, 192(1):489–504, 2012. doi:10.1007/s11856-012-0039-7.

56 Shyam Narayanan and Jelani Nelson. Optimal terminal dimensionality reduction in euclidean space. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 1064–1069. ACM, 2019. doi:10.1145/3313276.3316307.

57 Amir Nayyeri and Benjamin Raichel. Reality distortion: Exact and approximate algorithms for embedding into the line. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 729–747. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.50.

58 Anastasios Sidiropoulos, Mihai Badoiu, Kedar Dhamdhere, Anupam Gupta, Piotr Indyk, Yuri Rabinovich, Harald Räcke, and R. Ravi. Approximation algorithms for low-distortion embeddings into low-dimensional spaces. *SIAM J. Discret. Math.*, 33(1):454–473, 2019. doi:10.1137/17M1113527.

59 Anastasios Sidiropoulos, Dingkang Wang, and Yusu Wang. Metric embeddings with outliers. In *ACM-SIAM Symposium on Discrete Algorithms*, 2015.

60 Bang Ye Wu, Giuseppe Lancia, Vineet Bafna, Kun-Mao Chao, R. Ravi, and Chuan Yi Tang. A polynomial-time approximation scheme for minimum routing cost spanning trees. *SIAM Journal on Computing*, 29(3):761–778, 2000. doi:10.1137/S009753979732253X.

Tight Approximation Results for Matroid Optimization with a Linear Constraint

Ilan Doron-Arad
MIT, Cambridge, MA, USA

Hadas Shachnai
Technion – Israel Institute of Technology, Haifa, Israel

Gilad Shmerler
Technion – Israel Institute of Technology, Haifa, Israel

Abstract

We study the following class of *matroid optimization problems with a linear constraint* ($\mathcal{P}$-MOL). Given a matroid $\mathcal{M} = (E, \mathcal{I})$, two nonnegative weight functions $v, w : E \to \mathbb{R}_{\geq 0}$, and a threshold $L \in \mathbb{R}_{\geq 0}$, find $\mathsf{opt}\, v(S)$ where S is either an independent set or a base of $\mathcal{M}$ satisfying a budget-type constraint: $w(S) \leq L$ or $w(S) \geq L$, and $\mathsf{opt} \in \{\min, \max\}$. $\mathcal{P}$-MOL provides a unified representation for a broad family of NP-hard optimization problems, including BUDGETED MATROID INDEPENDENT SET, CONSTRAINED MINIMUM-BASIS, and KNAPSACK-COVER variants with a matroid constraint. Also, it naturally extends to multiple matroid constraints. In particular, we consider the MATROID INTERSECTION COVER (MIC) problem, where feasibility is defined by the common independent sets of two matroids and one seeks minimum $v(S)$ subject to $w(S) \geq L$.

Our main result is a unified *efficient polynomial-time approximation scheme* (EPTAS) for all nontrivial $\mathcal{P}$-MOL variants, obtained by generalizing a technique of Hassin and Levin (SIAM J. Comput., 2004) for solving the CONSTRAINED MINIMUM SPANNING TREE problem. Specifically, for any fixed $\varepsilon > 0$, we present an algorithm running in time $|E|^{O(1)} \cdot (1/\varepsilon^2)^{O(1/\varepsilon)}$ that outputs a feasible solution S whose value is at most $(1+\varepsilon)\mathsf{OPT}$ for minimization variants and at least $(1-\varepsilon)\mathsf{OPT}$ for maximization variants. This resolves the complexity status of all members of $\mathcal{P}$-MOL, as none of these problems admits a *fully* polynomial-time approximation scheme (Doron-Arad, Kulik and Shachnai, ICALP'24). Finally, we separate the $\mathcal{P}$-MOL family from its extension to matroid intersection. We show that an EPTAS is unlikely to exist for the matroid intersection variant of $\mathcal{P}$-MOL under a covering constraint, whereas an EPTAS is known to exist under a budget constraint. This highlights a qualitative difference between these two types of linear constraints that does not arise in the single-matroid setting.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Matroids, Knapsack, Linear Constraints, EPTAS, Budgeted Optimization, MOL Problems

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.18

Category APPROX

1 Introduction

Matroids offer a unifying abstraction for "independence" in combinatorial structures: they capture linear independence in vector spaces, forests in graphs, and many other systems constrained by an exchange principle. A *matroid* is a set system $(E, \mathcal{I})$, where E is a finite set and $\mathcal{I} \subseteq 2^E$ are the *independent sets (IS)* such that (i) $\emptyset \in \mathcal{I}$, (ii) for all $A \in \mathcal{I}$ and $B \subseteq A$ it holds that $B \in \mathcal{I}$, and (iii) for all $A, B \in \mathcal{I}$ where $|A| > |B|$, there is $e \in A \setminus B$ such that $B \cup \{e\} \in \mathcal{I}$.[1] A broad range of classical problems can be cast as *matroid*

[1] Properties (ii) and (iii) are known, respectively, as *hereditary property*, and *exchange property*.

© Ilan Doron-Arad, Hadas Shachnai, and Gilad Shmerler;
licensed under Creative Commons License CC-BY 4.0

optimization: one seeks a minimum-cost or maximum-value independent set or basis of a matroid. Canonical examples include minimum spanning trees, maximum-weight forests, and a variety of assignment and scheduling problems.

In many applications, however, matroid feasibility alone is insufficient. Solutions must also satisfy an additional *budget-type* constraint, such as an upper bound on total length or cost, or a lower bound on total coverage. Adding even a single linear constraint of this form typically turns an otherwise polynomial-time matroid optimization problem into an NP-hard one, and has motivated a long line of work on approximation algorithms.

1.1 The $\mathcal{P}$-MOL Class and Our Framework

We start by defining the class of problems studied in this paper. Let $\mathcal{M} = (E, \mathcal{I})$ be a matroid, and let $v, w : E \to \mathbb{R}_{\geq 0}$ be two nonnegative functions. The value v_e contributes to the main objective, and w_e contributes to a budget constraint. For the matroid $\mathcal{M} = (E, \mathcal{I})$ we define $\mathsf{IS}(\mathcal{M}) := \mathcal{I}$ and $\mathsf{bases}(\mathcal{M}) := \{ S \in \mathcal{I} : |S| = \mathsf{rank}(\mathcal{M}) \}$, where $\mathsf{rank}(\mathcal{M}) := \max\{ |T| : T \in \mathcal{I} \}$ is the rank of $\mathcal{M}$, i.e., the maximum cardinality of an independent set.

We are interested in three design choices:

- **Objective direction.** We may want to minimize or maximize $v(S)$.[2]
- **Matroid family.** We may optimize over independent sets or bases of $\mathcal{M}$.
- **Budget direction.** The budget constraint may take the form $w(S) \leq L$ or $w(S) \geq L$.

We encode these choices by three parameters

$$\mathsf{opt} \in \{\min, \max\}, \qquad \mathcal{F} \in \{\mathsf{IS}, \mathsf{bases}\}, \qquad \triangleright \in \{\leq, \geq\}.$$

Given a threshold $L \geq 0$, we consider the optimization problem

$$\mathsf{opt}\ v(S) \qquad \text{s.t.} \qquad S \in \mathcal{F}, \quad w(S)\ \triangleright\ L. \tag{1}$$

We refer to (1) as a $\mathcal{P}$-MOL (Matroid Optimization with a Linear constraint) problem, with the descriptor $\mathcal{P} = (\mathsf{opt}, \mathcal{F}, \triangleright)$. An instance of a $\mathcal{P}$-MOL problem is specified by a tuple $\mathcal{A}_{\mathcal{P}} = (E, \mathcal{I}, v, w, L)$.

For a fixed descriptor $\mathcal{P}$, we denote the optimal value of (1) by $\mathsf{OPT}_{\mathcal{P}}$, or simply OPT when the context is clear:

$$\mathsf{OPT}_{\mathcal{P}}\ :=\ \mathsf{opt}\big\{\, v(S) : S \in \mathcal{F}(\mathcal{M}),\ w(S) \triangleright L \,\big\},$$

This formulation captures, in a uniform way, several extensively studied problems. For example:

- $(\max, \mathsf{IS}, \leq)$ gives the BUDGETED MATROID INDEPENDENT SET problem, generalizing 0/1-KNAPSACK.
- $(\min, \mathsf{bases}, \leq)$ gives CONSTRAINED MINIMUM-BASIS problems, including CONSTRAINED MINIMUM SPANNING TREE when $\mathcal{M}$ is a graphic matroid.
- $(\min, \mathsf{IS}, \geq)$ corresponds to KNAPSACK-COVER WITH A MATROID.

Among the eight possible choices of $(\mathsf{opt}, \mathcal{F}, \triangleright)$, only $(\min, \mathsf{IS}, \leq)$ is trivial (its optimum is $S = \emptyset$); we focus on the remaining seven nontrivial variants.

The $\mathcal{P}$-MOL viewpoint also naturally extends beyond a single matroid. If one lets $\mathcal{F}$ denote the intersection of two matroids, then (1) becomes matroid-intersection optimization with a linear constraint. A notable example is the MATROID INTERSECTION COVER (MIC)

[2] For any vector $p = (p_e)_{e \in E}$ and any $S \subseteq E$ we use the shorthand $p(S) := \sum_{e \in S} p_e$.

problem: given matroids $\mathcal{M}_1 = (E, \mathcal{I}_1)$ and $\mathcal{M}_2 = (E, \mathcal{I}_2)$ and weights $v, w : E \to \mathbb{R}_{\geq 0}$, find a set $S \in \mathcal{I}_1 \cap \mathcal{I}_2$ with $w(S) \geq L$ minimizing $v(S)$. In our terminology, MIC aligns with a $(\min, \mathsf{intersection}, \geq)$-MOL formulation, with feasibility governed by $\mathcal{I} = \mathcal{I}_1 \cap \mathcal{I}_2$.

1.2 Our Contribution

We present (in Section 2) a unified approximation framework for $\mathcal{P}$-MOL problems by generalizing a technique of Hassin and Levin [20]. This allows to obtain an EPTAS for all non-trivial variants of $\mathcal{P}$. Specifically, for every fixed $\varepsilon > 0$ there exists an algorithm which, given an instance $\mathcal{A}_\mathcal{P}$, outputs a feasible solution S satisfying

$$\mathsf{opt} = \min \implies v(S) \leq (1+\varepsilon)\,\mathsf{OPT}_\mathcal{P}, \qquad \mathsf{opt} = \max \implies v(S) \geq (1-\varepsilon)\,\mathsf{OPT}_\mathcal{P}.$$

The running time is $h(1/\varepsilon) \cdot \mathrm{poly}(|\mathcal{A}_\mathcal{P}|)$ for an arbitrary function h. This resolves the complexity status of all members of $\mathcal{P}$-MOL, as none of these problems admits a *fully* polynomial-time approximation scheme (FPTAS) [15]. Table 1 presents the best known results for the $\mathcal{P}$-MOL class. Results obtained in this paper are marked in boldface.

We also consider the natural extension of $\mathcal{P}$-MOL variants to matroid intersection. For BUDGETED MATROID INTERSECTION, an EPTAS was presented in [12]. In contrast to our positive result for the $\mathcal{P}$-MOL class, where matroid optimization is over a *single* matroid constraint, we show that a unified EPTAS is unlikely to exist for *all* $\mathcal{P}$-MOL variants, already when the single matroid constraint is replaced by a *matroid intersection* constraint. Specifically, we prove (in Section 3) that MIC does not admit an EPTAS unless W[1] = FPT.

1.3 Techniques

A Unified EPTAS for $\mathcal{P}$-MOL

For BUDGETED MATROID INDEPENDENT SET, as well as for its extension to BUDGETED MATROID INTERSECTION, recent works [13, 12] give EPTASs based on the *representative-set* methodology. Roughly, it computes a small candidate pool that is guaranteed to contain the high-value elements of some near-optimal solution, and then recovers the near-optimum by exhaustive enumeration over this pool. However, this approach inherently (i) loses the profit of $O(1)$ elements, and (ii) may drop $O(1)$ elements to restore feasibility. Thus, a solution output by the representative set approach may be a non-basis for the given matroid. It may also be infeasible w.r.t. a covering constraint. Therefore, the EPTASs of [13, 12] cannot be extended to EPTASs for the *entire* $\mathcal{P}$-MOL class, which must handle both $\mathsf{IS}(\mathcal{M})$ and $\mathsf{bases}(\mathcal{M})$, and all four choices of $(\mathsf{opt}, \triangleright)$.

Instead, we generalize (in Section 2) a technique of Hassin and Levin [20] to obtain a unified EPTAS for all members of $\mathcal{P}$-MOL under a general matroid constraint. The following is a high-level overview of the algorithmic framework.

Given a constant-factor guarantee $\hat{V}$ for OPT, we bucket elements according to their values v_e. Elements with $v_e \leq \varepsilon\hat{V}$ are classified as *small*, while the remaining elements are partitioned into $O(1/\varepsilon^2)$ narrow buckets. We note that any optimal solution contains only $O(1/\varepsilon)$ *large* elements. This allows us to enumerate *configuration vectors* $\vec{k}$ that specify the number of selected elements from each bucket.

For a fixed configuration $\vec{k}$, we enforce these counts by intersecting the matroid with a partition matroid, yielding a matroid-intersection family $\mathcal{F}_{\vec{k}}$. We then use a Lagrangian relaxation of the linear constraint $w(S) \triangleright L$ and compute $\lambda^\star_{\vec{k}}$, an optimal value of the multiplier λ.

The main technical challenge is to convert the Lagrangian optimum into a feasible solution. At $\lambda^\star_{\vec{k}}$, the family $\mathcal{O}_{\vec{k}}$ of Lagrangian-optimal sets is *well connected*: any two of its members can be transformed into one another via a sequence of exchanges, each replacing at most one element per bucket. We compute sets $S^-, S^+ \in \mathcal{O}_{\vec{k}}$ that respectively minimize and maximize the signed violation of $w(S) \triangleright L$, and hence lie on opposite sides of feasibility. Traversing the one-per-bucket exchange path from S^- to S^+, we select the first feasible set $S^\star$.

Because elements within each bucket have nearly identical v-values, each exchange affects the objective value by at most the total *bucket spread* – that is, the difference between the maximum and minimum v-value within a bucket. Consequently, the first feasible set $S^\star$ yields an objective value within an additive $O(\varepsilon \mathsf{OPT})$ of the Lagrangian bound. Taking the best such $S^\star$ over all configurations yields an EPTAS.

We note that some variants in our descriptor-based formulation are related by standard reductions. Nevertheless, we keep the unified formulation for completeness and to make the scope of the algorithm explicit, since such reductions may not always preserve the approximation guarantees in a straightforward way. In particular, budget-type and covering-type constraints require careful treatment: the existence of an EPTAS for the budget setting under a matroid intersection constraint [12] does not, by itself, imply an analogous result for the covering setting. Our hardness result in Theorem 13 highlights this distinction. Therefore, we present the argument directly for the descriptor class $\mathcal{P} = (\mathsf{opt}, \mathcal{F}, \triangleright)$.

Hardness of MIC

We establish a parameterized lower bound for MATROID INTERSECTION COVER (MIC). Via a reduction from k-Clique, we show that MIC admits no EPTAS unless $W[1] = \text{FPT}$ (Theorem 13). The novelty of our result lies in the fact that this reduction rules out an *EPTAS*, whereas a similar construction of [15] is used only to exclude an *FPTAS*. Our result complements recent applications of related reductions to 3-MATROID INTERSECTION [16]. Together, our findings demonstrate that for matroid intersection, different linear constraints (cover versus budget) lead to markedly different algorithmic behavior, in contrast to the case of a single matroid.

Table 1 Our positive results for $\mathcal{P}$-MOL problems and the matching negative results of [15].

Problem Type	Opt	$\mathcal{F}$	$\triangleright$	Positive Result	Negative Result
Budgeted Matroid Independent Set	max	IS	$\leq$	EPTAS [13]	No FPTAS
Knapsack-Cover with a Matroid	min	IS	$\geq$	**EPTAS**	No FPTAS
Budgeted Matroid Basis	max	Bases	$\leq$	**EPTAS**	No FPTAS
Constrained Minimum Basis	min	Bases	$\leq$	**EPTAS** [3]	No FPTAS
Lower-Bounded Max Basis	max	Bases	$\geq$	**EPTAS**	No FPTAS
Lower-Bounded Min Basis	min	Bases	$\geq$	**EPTAS**	No FPTAS
Lower-Bounded Indep. Set	max	IS	$\geq$	**EPTAS**	No FPTAS

[3] Previously, an EPTAS was known for the special case of Constrained Minimum Basis in a *graphic* matroid [20].

1.4 Related Work

Knapsack and approximation schemes

The 0/1-Knapsack problem is a canonical benchmark for approximation schemes. Early work already established a fully polynomial-time approximation scheme [24], and dynamic-programming–based FPTASs and their implications have since been extensively studied [8]. More recent progress has focused on fine-grained analyses and improved time–approximation tradeoffs [4, 9].

Matroid and matching constraints with a single budget

Adding a matroid or matching feasibility constraint to 0/1-Knapsack gives rise to the well-studied problems of Budgeted Matching and Budgeted Matroid Intersection. A PTAS for both variants was obtained by Berger et al. [1]. Subsequent work strengthened these results to EPTAS guarantees: for Budgeted Matroid Independent Set [13], and for Budgeted Matching and Budgeted Matroid Intersection via representative sets [12].

While an FPTAS is ruled out for general instances of Budgeted Matroid Independent Set [15], FPTASs are known for several special cases, including Cardinality-Constrained Knapsack [5], Multiple-Choice Knapsack [22], and laminar matroids [14, 27]. Additional variants of the problem have also been studied, for example in [11, 10, 17].

$\mathcal{P}$-MOL problems have also been studied in the covering setting (as opposed to a budget constraint), in conjunction with a matroid feasibility constraint. For Lower-Bounded Minimum Independent Set, the best known result prior to this work was a PTAS, due to [6]. For constrained minimum spanning tree, a PTAS was given in [26], followed by an EPTAS in [20]. Furthermore, a multi-criteria FPTAS is known for this problem [21].

Multiple budgets

When multiple budget constraints are present, PTASs are known for variants with either matroid intersection or matching constraints, using techniques such as dependent randomized rounding [7] and structural properties of matroid polytopes [19]. On the other hand, even seemingly simple extensions face strong complexity barriers: for example, the Multidimensional Knapsack problem with just two budget constraints already rules out an EPTAS under standard complexity assumptions [23].

2 An EPTAS for the $\mathcal{P}$-MOL Class

Let $\mathcal{A}_{\mathcal{P}} = (E, \mathcal{I}, v, w, L)$ be an instance of $\mathcal{P}$-MOL with descriptor $\mathcal{P} = (\mathsf{opt}, \mathcal{F}, \triangleright)$ and fix $\varepsilon > 0$. Recall that $\mathcal{F}$ denotes the underlying combinatorial family: either all independent sets or all bases of $\mathcal{M}$. Our EPTAS combines a value-bucketing scheme with a Lagrangian relaxation applied to the intersection of $\mathcal{M}$ with a partition matroid (defined below). We begin by describing the bucketing and configuration step. Let $n := |E|$ denote the ground-set size.

2.1 Value Bucketing and Configuration Vectors

We first obtain a constant-factor estimate of OPT.

▶ **Lemma 1.** *Given an instance $\mathcal{A}_{\mathcal{P}}$, one can guess in $|E|^{O(1)}$ time a value $\hat{V}$ such that*

$$\frac{OPT}{2} \le \hat{V} \le OPT \quad \text{if } \mathsf{opt} = \max, \qquad \text{and} \qquad OPT \le \hat{V} \le 2OPT \quad \text{if } \mathsf{opt} = \min.$$

Proof. We obtain $\hat{V}$ by iterating over all powers of 2 in the domain $[0, v(E)]$. In one of these iterations we necessarily hit a value within a factor 2 of OPT, and we output this guess as $\hat{V}$. For simplicity of presentation, in the remainder we assume that we are given such a value $\hat{V}$ satisfying the stated 2-approximation guarantees.

The number of guesses is $O(\log v(E))$. Since $\mathsf{OPT} \le v(E) \le n\, v_{\max}$,

$$\log v(E) \le \log(n v_{\max}) = O(\log n + \log v_{\max}).$$

Let b be the bit-length of $v_{\max}$. If $b = \mathrm{poly}(n)$ we are done; otherwise b is exponential in n and we can solve the instance exactly by enumerating all 2^n subsets of E, which is still polynomial in $|E|$. ◀

We assume w.l.o.g. that $v_e \le 2\hat{V}$ for all $e \in E$ (any e with $v_e > 2\hat{V}$ cannot belong to an optimal solution, since all values are nonnegative and $\mathsf{OPT} \le 2\hat{V}$ by Lemma 1). We partition the ground set E into bins according to the values v_e. We first define the bin of *small* elements:

$$B_0 := \{e \in E : v_e \le \varepsilon \hat{V}\}.$$

All other elements are considered *large* and are further categorized into $T = \left\lceil \frac{2-\varepsilon}{\varepsilon^2} \right\rceil$ bins:

$$B_i := \left\{e \in E : (\varepsilon + (i-1)\varepsilon^2)\hat{V} < v_e \le (\varepsilon + i\varepsilon^2)\hat{V}\right\}, \qquad i = 1, \dots, T.$$

▶ **Lemma 2.** *In any optimal solution S^{opt}, at most $2/\varepsilon$ elements belong to $\bigcup_{i=1}^{T} B_i$.*

Proof. Suppose, for the sake of contradiction, that S^{opt} contains more than $2/\varepsilon$ elements from $B_1 \cup \cdots \cup B_T$. By the definition of these bins, every such element has value strictly larger than $\varepsilon\hat{V}$, and hence

$$v\big(S^{\mathsf{opt}}\big) \ge v\big(S^{\mathsf{opt}} \cap (B_1 \cup \cdots \cup B_T)\big) \;>\; \frac{2}{\varepsilon} \cdot \varepsilon\hat{V} \;=\; 2\hat{V}.$$

Hence $v(S^{\mathsf{opt}}) > 2\hat{V}$. If $\mathsf{opt} = \max$, then by Lemma 1 we have $\mathsf{OPT} \le 2\hat{V}$, contradicting $v(S^{\mathsf{opt}}) = \mathsf{OPT} > 2\hat{V}$. If $\mathsf{opt} = \min$, then Lemma 1 gives $\mathsf{OPT} \le \hat{V}$ and hence $\mathsf{OPT} \le 2\hat{V}$, again contradicting $v(S^{\mathsf{opt}}) = \mathsf{OPT} > 2\hat{V}$. Thus S^{opt} contains at most $2/\varepsilon$ elements from $\bigcup_{i=1}^{T} B_i$. ◀

We therefore restrict our attention to solutions with at most $2/\varepsilon$ large elements. We encode such solutions via a *configuration vector* $\vec{k} = (k_0, k_1, \dots, k_T)$, where k_i denotes the number of elements that should be chosen from bin B_i. We only consider vectors satisfying

$$\sum_{i=1}^{T} k_i \le \left\lceil \frac{2}{\varepsilon} \right\rceil, \qquad \sum_{i=0}^{T} k_i \le |E|. \tag{2}$$

For such a vector $\vec{k}$, a set $S \subseteq E$ *respects the configuration* $\vec{k}$ if $|S \cap B_i| \;=\; k_i$ for all $i = 0, \dots, T$.

▶ **Lemma 3.** *The number of distinct configuration vectors satisfying* (2) *is at most* $O\left(|E| \cdot \left(\frac{1}{\varepsilon^2}\right)^{2/\varepsilon}\right)$.

Proof. Let $m := \lceil 2/\varepsilon \rceil$. By (2), the number of valid tuples $(k_1, \ldots, k_T)$ corresponds to the number of non-negative integer solutions to the inequality $\sum_{i=1}^{T} k_i \leq m$. By introducing a slack variable $k^\star \geq 0$ such that $k^\star + \sum_{i=1}^{T} k_i = m$, this count is equivalent to the number of weak compositions of m into $T+1$ parts. Applying the stars and bars formula, we obtain exactly

$$\binom{(T+1)+m-1}{m} = \binom{T+m}{m}$$

configurations. For each fixed tuple, k_0 is bounded by $|E|$. For a fixed ε (and thus fixed m), the number of configurations is

$$|E| \cdot \binom{T+m}{m} \;\leq\; |E| \cdot (T+m)^m \;=\; O(|E| \cdot T^m) \;=\; O\left(|E| \cdot \left(\frac{1}{\varepsilon^2}\right)^{2/\varepsilon}\right). \qquad \blacktriangleleft$$

Let $\vec{k}^{\mathsf{opt}}$ denote the configuration vector of an optimal solution S^{opt}. By Lemma 2, $\vec{k}^{\mathsf{opt}}$ satisfies (2), hence it is among the vectors considered by our algorithm.

2.2 Partition Matroid and Restricted Family

Fix a configuration vector $\vec{k} = (k_0, \ldots, k_T)$ satisfying (2), and let $K := \sum_{i=0}^{T} k_i$. We can enforce the cardinality of $\mathcal{M}$ by considering its truncation to rank K, denoted $\mathcal{M}_{\leq K}$, whose independent sets are $\mathcal{I}_{\leq K} = \mathsf{IS}(\mathcal{M}_{\leq K}) = \{S \in \mathcal{I} : |S| \leq K\}$.

Next, we encode the bin-wise upper bounds prescribed by $\vec{k}$ via a partition matroid. Let $B_0, \ldots, B_T$ be the bins (a partition of E). Define the partition matroid $\mathcal{M}^P_{\vec{k}}$ on E by

$$S \in \mathcal{I}^P_{\vec{k}} = \mathsf{IS}(\mathcal{M}^P_{\vec{k}}) \quad \Longleftrightarrow \quad |S \cap B_i| \leq k_i \;\text{ for all } i = 0, \ldots, T.$$

We restrict the $\mathcal{P}$-MOL problem to sets that satisfy both matroid constraints and have exactly K elements:

$$\mathcal{F}_{\vec{k}} \;:=\; \{\, S \subseteq E :\; S \in \mathcal{I}_{\leq K} \cap \mathcal{I}^P_{\vec{k}} \text{ and } |S| = K \,\}.$$

If $K > \mathsf{rank}(\mathcal{M})$, then $\mathcal{F}_{\vec{k}} = \emptyset$ (since no independent set can have size K); in the algorithm we simply skip such configurations.

Equivalently, $\mathcal{F}_{\vec{k}}$ is the family of common bases of $\mathcal{M}_{\leq K}$ and $\mathcal{M}^P_{\vec{k}}$ (since in each matroid, a set is a base iff it is independent and has size equal to the matroid rank, here K). If the original feasible family is $\mathcal{F} = \mathsf{IS}(\mathcal{M})$, then the above definition applies for any $K \leq \mathsf{rank}(\mathcal{M})$. If instead $\mathcal{F} = \mathsf{bases}(\mathcal{M})$, then we must have $K = \mathsf{rank}(\mathcal{M})$; otherwise no set of size K can be a base of $\mathcal{M}$. Accordingly, in the bases variant we only consider configuration vectors with $K = \mathsf{rank}(\mathcal{M})$ (and for other K we take $\mathcal{F}_{\vec{k}} = \emptyset$).

Finally, note that membership in $\mathcal{F}_{\vec{k}}$ forces the bin constraints of $\mathcal{M}^P_{\vec{k}}$ to be tight. Indeed, for any $S \in \mathcal{F}_{\vec{k}}$, we have

$$|S \cap B_i| \leq k_i \;\; \forall i \in \{0, \ldots, T\} \quad \text{and} \quad \sum_{i=0}^{T} |S \cap B_i| = |S| = K = \sum_{i=0}^{T} k_i,$$

implying that $|S \cap B_i| = k_i$ for all i. In particular, every $S \in \mathcal{F}_{\vec{k}}$ is a base of the partition matroid $\mathcal{M}^P_{\vec{k}}$ and also a base of the truncated matroid $\mathcal{M}_{\leq K}$ since $|S| = K$.

For $\mathsf{opt} = \max$, let $\mathsf{OPT}_{\vec{k}}$ denote the maximum of $v(S)$ over all $S \in \mathcal{F}_{\vec{k}}$ satisfying $w(S) \triangleright L$; for $\mathsf{opt} = \min$ we define $\mathsf{OPT}_{\vec{k}}$ analogously as a minimum. If $\vec{k} = \vec{k}^{\mathsf{opt}}$ is the configuration of an optimal solution, then clearly $\mathsf{OPT}_{\vec{k}^{\mathsf{opt}}} = \mathsf{OPT}$.

2.3 Lagrangian Relaxation

Returning to our problem, we must satisfy both the matroid-intersection constraint and the linear constraint: $w(S) \triangleright L$. Since the linear constraint is typically the harder one to handle directly, we apply *Lagrangian relaxation* to incorporate it into the objective via a penalty term.

To express the different orientations in a unified form, we define two sign parameters:

$$s_{\mathsf{opt}} = \begin{cases} +1 & \text{if } \mathsf{opt} = \max, \\ -1 & \text{if } \mathsf{opt} = \min, \end{cases} \qquad s_{\triangleright} = \begin{cases} +1 & \text{if } \triangleright = \geq, \\ -1 & \text{if } \triangleright = \leq . \end{cases}$$

For $\lambda \geq 0$ and $S \subseteq E$, we define the Lagrangian objective

$$\phi_\lambda(S) \;:=\; v(S) + s_{\mathsf{opt}} s_{\triangleright} \lambda\big(w(S) - L\big).$$

The corresponding Lagrangian relaxation value for some fixed configuration vector $\vec{k}$ is

$$V_{\vec{k}}(\lambda) \;:=\; \mathsf{opt}_{S \in \mathcal{F}_{\vec{k}}}\, \phi_\lambda(S). \tag{3}$$

Finally, we define the optimal value of the Lagrangian relaxation by

$$\mathrm{LR}_{\vec{k}} := \begin{cases} \min\limits_{\lambda \geq 0} V_{\vec{k}}(\lambda) & \text{if } \mathsf{opt} = \max, \\ \max\limits_{\lambda \geq 0} V_{\vec{k}}(\lambda) & \text{if } \mathsf{opt} = \min . \end{cases}$$

▶ **Lemma 4.** *For every configuration vector $\vec{k}$, the function $V_{\vec{k}} : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is piecewise linear. Moreover, if $\mathsf{opt} = \max$, then $V_{\vec{k}}$ is convex, and if $\mathsf{opt} = \min$, then $V_{\vec{k}}$ is concave.*

Proof. Fix $\vec{k}$ and consider any $S \in \mathcal{F}_{\vec{k}}$. The function $\phi_\lambda(S)$ is affine in λ. Therefore, $V_{\vec{k}}(\lambda)$ is the pointwise optimum of finitely many affine functions (one for each $S \in \mathcal{F}_{\vec{k}}$), and hence $V_{\vec{k}}$ is piecewise linear.

Moreover, if $\mathsf{opt} = \max$ then $V_{\vec{k}}$ is the pointwise maximum of affine functions and hence convex; if $\mathsf{opt} = \min$ then $V_{\vec{k}}$ is the pointwise minimum of affine functions and hence concave. ◀

▶ **Lemma 5.** *For every configuration vector $\vec{k}$, if $\mathsf{opt} = \max$, then $\mathrm{LR}_{\vec{k}} \geq \mathit{OPT}_{\vec{k}}$, and if $\mathsf{opt} = \min$, then $\mathrm{LR}_{\vec{k}} \leq \mathit{OPT}_{\vec{k}}$.*

Proof. Let S be any feasible solution with configuration $\vec{k}$, i.e., $S \in \mathcal{F}_{\vec{k}}$ and $w(S) \triangleright L$. Then $s_{\triangleright}\big(w(S) - L\big) \geq 0$. For any $\lambda \geq 0$,

$$s_{\mathsf{opt}}\, \phi_\lambda(S) = s_{\mathsf{opt}}\, v(S) + s_{\triangleright}\, \lambda\big(w(S) - L\big) \;\geq\; s_{\mathsf{opt}}\, v(S).$$

Optimizing over all such S yields $s_{\mathsf{opt}} V_{\vec{k}}(\lambda) \geq s_{\mathsf{opt}} \mathsf{OPT}_{\vec{k}}$. If $\mathsf{opt} = \max$ (so $s_{\mathsf{opt}} = +1$), this implies $V_{\vec{k}}(\lambda) \geq \mathsf{OPT}_{\vec{k}}$ for all λ, hence $\mathrm{LR}_{\vec{k}} = \min_{\lambda \geq 0} V_{\vec{k}}(\lambda) \geq \mathsf{OPT}_{\vec{k}}$. The minimization case is analogous. ◀

We now argue that the Lagrangian relaxation (3) can be solved efficiently.

▶ **Lemma 6.** *For every configuration vector $\vec{k}$ and any $\lambda \geq 0$, the value $V_{\vec{k}}(\lambda)$ and a set $S_{\vec{k},\lambda} \in \mathcal{F}_{\vec{k}}$ attaining this value can be computed in polynomial time. Moreover, a multiplier $\lambda^\star_{\vec{k}}$ satisfying $V_{\vec{k}}(\lambda^\star_{\vec{k}}) = \mathrm{LR}_{\vec{k}}$ can be computed in polynomial time.*

Proof. Fix a configuration vector $\vec{k}$ and $\lambda \geq 0$, and define the modified weight function v^λ as follows

$$v_e^\lambda := v_e + s_{\mathsf{opt}} s_\triangleright \lambda w_e \quad \text{for all } e \in E.$$

Then, for any $S \subseteq E$, we can write

$$\phi_\lambda(S) = v(S) + s_{\mathsf{opt}} s_\triangleright \lambda\big(w(S) - L\big) = \sum_{e \in S} v_e^\lambda \; - \; s_{\mathsf{opt}} s_\triangleright \lambda L.$$

Hence, for fixed λ, optimizing ϕ_λ over $S \in \mathcal{F}_{\vec{k}}$ is equivalent (up to an additive constant independent of S) to solving the following optimization problem

$$\mathsf{opt}_{S \in \mathcal{F}_{\vec{k}}} \sum_{e \in S} v_e^\lambda.$$

For both $\mathcal{F} \in \{\mathsf{IS}, \mathsf{bases}\}$, the restriction to $\mathcal{F}_{\vec{k}}$ imposes a fixed cardinality constraint $|S| = K$ (where $K = \sum_i k_i$, and in the bases case necessarily $K = \mathsf{rank}(\mathcal{M})$, otherwise $\mathcal{F}_{\vec{k}} = \emptyset$). Consequently, the problem of optimizing a linear objective over $\mathcal{F}_{\vec{k}}$ is a cardinality-constrained weighted matroid intersection instance between $\mathcal{M}$ (or $\mathcal{M}_{\leq K}$) and the partition matroid $\mathcal{M}_{\vec{k}}^P$. This problem is solvable in polynomial time by the algorithm of Brezovec et al. [2]. Therefore, for any fixed $\vec{k}$ and $\lambda \geq 0$, we can compute $V_{\vec{k}}(\lambda)$ in polynomial time.

It remains to find a multiplier $\lambda_{\vec{k}}^\star \geq 0$ such that $V_{\vec{k}}(\lambda_{\vec{k}}^\star) = \mathrm{LR}_{\vec{k}}$. Since the objective weights v_e^λ depend linearly on λ, we can apply Megiddo's parametric-search framework [25] to compute such an optimal multiplier $\lambda_{\vec{k}}^\star$ in polynomial time. ◀

Let $\lambda_{\vec{k}}^\star$ be such an optimal multiplier, and define the family of Lagrangian–optimal solutions

$$\mathcal{O}_{\vec{k}} := \{\, S \in \mathcal{F}_{\vec{k}} : \phi_{\lambda_{\vec{k}}^\star}(S) = \mathrm{LR}_{\vec{k}} \,\}.$$

For every $S \in \mathcal{O}_{\vec{k}}$ we have $\phi_{\lambda_{\vec{k}}^\star}(S) = \mathrm{LR}_{\vec{k}}$, which can be rewritten as

$$v(S) = \mathrm{LR}_{\vec{k}} - s_{\mathsf{opt}} s_\triangleright \, \lambda_{\vec{k}}^\star\big(w(S) - L\big). \tag{4}$$

We next consider the degenerate edge case in which $\lambda_{\vec{k}}^\star = 0$. In this case, the Lagrangian objective reduces to the original objective over $\mathcal{F}_{\vec{k}}$. Let

$$S_{\vec{k}}^0 \in \operatorname*{arg\,opt}_{S \in \mathcal{F}_{\vec{k}}} v(S)$$

be an unconstrained optimum chosen to maximize $s_\triangleright(w(S) - L)$ among all unconstrained optima. If $S_{\vec{k}}^0$ is feasible, we add it directly to the candidate pool. If no unconstrained optimum is feasible, this configuration contributes no candidate in the case $\lambda_{\vec{k}}^\star = 0$. Henceforth, we assume that $\lambda_{\vec{k}}^\star > 0$.

2.4 Spread and Adjacency

We introduce a bound on the change in $v(\cdot)$ under a one-per-bin exchange, and then prove an adjacency property of $\mathcal{O}_{\vec{k}}$ showing that any two solutions in $\mathcal{O}_{\vec{k}}$ are connected by a sequence of such exchanges. For each bin B_i define its *spread*

$$\Delta_i \; := \; \max_{e \in B_i} v_e \; - \; \min_{e \in B_i} v_e,$$

with the convention that $\Delta_i = 0$ if $B_i = \emptyset$. For a configuration vector $\vec{k}$, define the total spread

$$\Sigma_\Delta(\vec{k}) := \sum_{i:\, k_i > 0} \Delta_i.$$

▶ **Lemma 7.** *Fix a configuration vector $\vec{k}$ satisfying (2).*

1. *If $S, S' \in \mathcal{F}_{\vec{k}}$ differ by exchanging at most one element per bin, then $|v(S) - v(S')| \le \Sigma_\Delta(\vec{k})$.*
2. *It holds that $\Sigma_\Delta(\vec{k}) \le 8\varepsilon\, \mathit{OPT}$.*

Proof. *Part* (1). Fix a bin B_i with $k_i > 0$. By assumption, passing from S to S' removes at most one element from B_i and therefore (to maintain $|S' \cap B_i| = |S \cap B_i|$, since $S, S' \in \mathcal{F}_{\vec{k}}$) it also adds at most one element to B_i. Hence the contribution of bin B_i to the value can change by at most Δ_i, i.e., $|v(S \cap B_i) - v(S' \cap B_i)| \le \Delta_i$. Summing over all bins with $k_i > 0$ yields

$$|v(S) - v(S')| \le \sum_{i:\, k_i > 0} |v(S \cap B_i) - v(S' \cap B_i)| \le \sum_{i:\, k_i > 0} \Delta_i = \Sigma_\Delta(\vec{k}).$$

Part (2). By the bucketing construction, $\Delta_0 \le \varepsilon \hat{V}$ and $\Delta_i \le \varepsilon^2 \hat{V}$ for all $i \ge 1$. Moreover, (2) implies that the number of indices $i \ge 1$ with $k_i > 0$ is at most $\lceil 2/\varepsilon \rceil$. Therefore,

$$\Sigma_\Delta(\vec{k}) = \Delta_0 \cdot \mathbf{1}[k_0 > 0] + \sum_{i \ge 1:\, k_i > 0} \Delta_i \le \varepsilon \hat{V} + \left\lceil \frac{2}{\varepsilon} \right\rceil \cdot \varepsilon^2 \hat{V} \le \varepsilon \hat{V} + 3\varepsilon \hat{V} \le 4\varepsilon \hat{V}.$$

Finally, by Lemma 1, we have $\hat{V} \le \mathsf{OPT}$ when $\mathsf{opt} = \max$, and $\hat{V} \le 2\mathsf{OPT}$ when $\mathsf{opt} = \min$; thus, in both cases $\Sigma_\Delta(\vec{k}) \le 8\varepsilon\, \mathsf{OPT}$. ◀

We next establish the adjacency property of $\mathcal{O}_{\vec{k}}$, namely that any two sets in $\mathcal{O}_{\vec{k}}$ can be connected by a sequence of exchanges that swaps at most one element per bin. Our proof uses the following theorem of Brezovec et al. [3].

▶ **Lemma 8** (Theorem 4 from [3]). *Let $\mathcal{M}_1 = (E, \mathcal{I}_1)$ be a matroid and let $\mathcal{M}_2 = (E, \mathcal{I}_2)$ be a partition matroid on the same ground set E, given by a partition $E = B_0 \cup \cdots \cup B_T$ and capacities $(k_0, \ldots, k_T)$. Let $c : E \to \mathbb{R}$ be a weight function, and let S be a common base of $\mathcal{M}_1$ and $\mathcal{M}_2$.*

Define the directed bipartite exchange graph *$B_S = (U \cup V, A)$ by $U := S, V := E \setminus S$, (i.e., one vertex for each element), and for each $e \in U$ and $f \in V$ define arcs as follows:*

- *if $S - e + f \in \mathcal{I}_1$, add an arc (e, f) with cost $c(f) - c(e)$;*
- *if $S - e + f \in \mathcal{I}_2$, add an arc (f, e) with cost 0.*

Let D_S be the directed graph obtained from B_S by compressing, for each $i \in \{0, \ldots, T\}$, all vertices in $V \cap B_i$ into a single vertex v_i, and then deleting parallel arcs while keeping, for each ordered pair of vertices, only the arc of minimum length.

For the graph D_S the following hold:

- *There are no negative cycles in D_S if and only if S is optimal for the weighted matroid intersection instance.*
- *If D_S contains a directed cycle C of negative total length, then there exists a common base T of $\mathcal{M}_1$ and $\mathcal{M}_2$ that is obtained from S by exchanging elements along C and satisfies $c(T) < c(S)$.*

Using this lemma, we obtain the following exchange-path guarantee.

▶ **Lemma 9.** *For any $S, S' \in \mathcal{O}_{\vec{k}}$, we can find in polynomial time a sequence of solutions $S = S_0, S_1, \ldots, S_\ell = S'$ such that:*

1. *$S_j \in \mathcal{O}_{\vec{k}}$ for all $j = 0, \ldots, \ell$;*
2. *For each $j = 0, \ldots, \ell - 1$ and for each bin B_i,*

$$|(S_j \setminus S_{j+1}) \cap B_i| = |(S_{j+1} \setminus S_j) \cap B_i| \leq 1.$$

Proof. Fix a configuration vector $\vec{k}$, and let $v^* = v^{\lambda^*_{\vec{k}}}$. W.l.o.g., assume that $\mathcal{O}_{\vec{k}}$ consists of *minimum*-cost bases with respect to v^* (otherwise, replace v^* by $-v^*$).

We prove the statement by induction on $x := |S' \setminus S|$. The case $x = 0$ is trivial. Assume that $x \geq 1$ and the lemma holds for every pair $T, T' \in \mathcal{O}_{\vec{k}}$ satisfying $|T' \setminus T| < x$. We prove the claim for the given pair S, S' with $|S' \setminus S| = x$.

Fix $\delta > 0$ and define $v' : E \to \mathbb{R} \cup \{+\infty\}$ by

$$v'(e) = \begin{cases} v^*(e) + \delta, & e \in S \setminus S', \\ v^*(e) - \delta, & e \in S' \setminus S, \\ v^*(e), & e \in S \cap S', \\ +\infty, & e \notin S \cup S'. \end{cases}$$

Then $v'(S) = v^*(S) + x\delta$ and $v'(S') = v^*(S') - x\delta$, and hence $v'(S') = v^*(S') - x\delta < v^*(S) + x\delta = v'(S)$, where we used $v^*(S) = v^*(S')$ since $S, S' \in \mathcal{O}_{\vec{k}}$. To avoid choosing an explicit infinitesimal δ, we implement this perturbation lexicographically. Let U be an integer upper bound on $\sum_{e \in S} |v^*(e)|$ over all $S \in \mathcal{F}_{\vec{k}}$ (e.g., $U := K \cdot \max_{e \in E} |v^*(e)|$), and set $M := 2U + 1$. Define an auxiliary weight function $\tilde{v} : E \to \mathbb{R}$ by

$$\tilde{v}(e) = \begin{cases} M \cdot v^*(e) + 1, & e \in S \setminus S', \\ M \cdot v^*(e) - 1, & e \in S' \setminus S, \\ M \cdot v^*(e), & e \in S \cap S', \\ M \cdot v^*(e) + M^2 + K, & e \notin S \cup S'. \end{cases}$$

We claim that $\tilde{v}(S') < \tilde{v}(S)$. Indeed, since $S, S' \in \mathcal{O}_{\vec{k}}$ we have $v^*(S) = v^*(S')$. Moreover, letting $x := |S' \setminus S| = |S \setminus S'|$, the definition of $\tilde{v}$ implies

$$\tilde{v}(S) = \sum_{e \in S} \tilde{v}(e) = M \sum_{e \in S} v^*(e) + |S \setminus S'| = M\, v^*(S) + x,$$

and similarly

$$\tilde{v}(S') = \sum_{e \in S'} \tilde{v}(e) = M \sum_{e \in S'} v^*(e) - |S' \setminus S| = M\, v^*(S') - x.$$

Therefore,

$$\tilde{v}(S) - \tilde{v}(S') = M\big(v^*(S) - v^*(S')\big) + 2x = 2x > 0,$$

and hence $\tilde{v}(S') < \tilde{v}(S)$.

Since the weights $v^*(e)$ may be rational, we first scale them to integers. Let D be a common denominator of $\{v^*(e) : e \in E\}$, and define $\bar{v}^*(e) := D \cdot v^*(e)$ for all $e \in E$. Then $\bar{v}^*(e) \in \mathbb{Z}$ for all e, and for any two common bases T_1, T_2 we have $\bar{v}^*(T_1) - \bar{v}^*(T_2) \in \mathbb{Z}$.

We redefine $\tilde{v}$ using $\bar{v}^*$ as follows. Let $U := K \cdot \max_{e \in E} |\bar{v}^*(e)|$ and set $M := 2U + 1$. Define $\tilde{v} : E \to \mathbb{Z}$ by $\tilde{v}(e) = M \cdot \bar{v}^*(e) + 1$ for $e \in S \setminus S'$, $\tilde{v}(e) = M \cdot \bar{v}^*(e) - 1$ for $e \in S' \setminus S$, $\tilde{v}(e) = M \cdot \bar{v}^*(e)$ for $e \in S \cap S'$, and $\tilde{v}(e) = M \cdot \bar{v}^*(e) + M^2 + 2MU + 2K$ for $e \notin S \cup S'$.

Our redefinition preserves the computation for S and S'. In particular, using $\bar{v}^*(S) = \bar{v}^*(S')$ we have $\tilde{v}(S) = M \cdot \bar{v}^*(S) + x$ and $\tilde{v}(S') = M \cdot \bar{v}^*(S') - x$, so $\tilde{v}(S') < \tilde{v}(S)$.

▷ Claim 10. Any $\tilde{v}$-minimum common base is (i) a v^*-minimum common base (and hence belongs to $\mathcal{O}_{\vec{k}}$), and (ii) consists solely of elements from $S \cup S'$.

Proof. We first show that minimizing $\tilde{v}$ enforces minimization of v^* (lexicographically). Fix any common base $T \in \mathcal{F}_{\vec{k}}$. Write $\tilde{v}(T) = M \cdot \bar{v}^*(T) + \tau(T)$, where $\tau(T)$ is the additive tie-breaking term induced by the ± 1 offsets (and 0 offsets) in the definition of $\tilde{v}$. Since $|T| = K$, we have the uniform bound $|\tau(T)| \le K$ whenever $T \subseteq S \cup S'$. Moreover, if T contains some element $e \notin S \cup S'$, then

$$\tilde{v}(T) \ge M \cdot \bar{v}^*(T) - K + (M^2 + 2MU + 2K) \ge -MU - K + (M^2 + 2MU + 2K) = M^2 + MU + K.$$

On the other hand, for every common base $T \subseteq S \cup S'$ we have

$$\tilde{v}(T) = M \cdot \bar{v}^*(T) + \tau(T) \le MU + K.$$

Therefore any $\tilde{v}$-minimum common base must be contained in $S \cup S'$.

Now let $T_1, T_2 \subseteq S \cup S'$ be any two common bases. Then

$$\tilde{v}(T_1) - \tilde{v}(T_2) = M\big(\bar{v}^*(T_1) - \bar{v}^*(T_2)\big) + \big(\tau(T_1) - \tau(T_2)\big),$$

and $|\tau(T_1) - \tau(T_2)| \le 2K$. Choose U so that $U \ge K \cdot \max_{e \in E} |\bar{v}^*(e)|$, and set $M := 2U + 1$ as above. Then $M > 2K$. Moreover, since $\bar{v}^*(T_1), \bar{v}^*(T_2) \in \mathbb{Z}$, if $\bar{v}^*(T_1) > \bar{v}^*(T_2)$ then $\bar{v}^*(T_1) - \bar{v}^*(T_2) \ge 1$, and hence

$$\tilde{v}(T_1) - \tilde{v}(T_2) = M\big(\bar{v}^*(T_1) - \bar{v}^*(T_2)\big) + \big(\tau(T_1) - \tau(T_2)\big) \ge M - 2K > 0.$$

Therefore $\tilde{v}(T_1) > \tilde{v}(T_2)$. Consequently, among all common bases contained in $S \cup S'$, every $\tilde{v}$-minimum base is also v^*-minimum. Moreover, as shown above, if a common base T contains an element outside $S \cup S'$, then $\tilde{v}(T) \ge M^2 + MU + K$, whereas every common base $T \subseteq S \cup S'$ satisfies $\tilde{v}(T) \le MU + K$. Hence no $\tilde{v}$-minimum common base uses elements outside $S \cup S'$. Therefore, any $\tilde{v}$-minimum common base belongs to $\mathcal{O}_{\vec{k}}$ and is contained in $S \cup S'$. ◁

Apply Lemma 8 to the pair $(\mathcal{M}_{\le K}, \mathcal{M}^P_{\vec{k}})$, the weight function $\tilde{v}$, and the current base S. Since $\tilde{v}(S') < \tilde{v}(S)$, the base S is not $\tilde{v}$-optimal, so the compressed graph D_S contains a negative cycle. Let C be a negative directed cycle in D_S with the minimum number of arcs. Lemma 8 yields a common base S'' obtained from S by exchanging along C such that $\tilde{v}(S'') < \tilde{v}(S)$. By the choice of $\tilde{v}(e)$ for $e \notin S \cup S'$, we have $S'' \subseteq S \cup S'$.

Since D_S is bipartite, any directed cycle C alternates between vertices $e \in S$ and contracted bin vertices v_i. We chose C with the minimum number of arcs, hence C cannot visit the same v_i twice; otherwise C would split into two directed cycles, and at least one of them would still have negative total length but fewer arcs, contradicting minimality. Thus, C visits each bin at most once, so the corresponding exchange inserts at most one element from each bin B_i, i.e. $|(S'' \setminus S) \cap B_i| \le 1$. Finally, S and S'' are both bases of the partition matroid $\mathcal{M}^P_{\vec{k}}$, so in each bin B_i an insertion must be matched by a deletion from the same bin in order to keep $|S \cap B_i| = k_i$. Hence, the numbers of inserted and deleted elements match in every bin:

$$|(S \setminus S'') \cap B_i| = |(S'' \setminus S) \cap B_i| \le 1 \qquad \forall i \in \{0, \dots, T\}.$$

Since $S \in \mathcal{O}_{\vec{k}}$ is v^*-minimum, we have $v^*(T) \ge v^*(S)$ for every common base T. In particular, for every common base $T \subseteq S \cup S'$ we have $\bar{v}^*(T) \ge \bar{v}^*(S)$. Now $S'' \subseteq S \cup S'$ and $\tilde{v}(S'') < \tilde{v}(S)$. If $\bar{v}^*(S'') > \bar{v}^*(S)$ then $\tilde{v}(S'') - \tilde{v}(S) \ge M - 2K > 0$, a contradiction. Hence $\bar{v}^*(S'') = \bar{v}^*(S)$, i.e., $v^*(S'') = v^*(S)$, and therefore $S'' \in \mathcal{O}_{\vec{k}}$.

Finally, since the inserted elements come from $E \setminus S$ and, by the definition of $\tilde{v}$, must lie in $S' \setminus S$, we add at least one element of $S' \setminus S$; thus $|S' \setminus S''| < |S' \setminus S| = x$.

The induction hypothesis applies to the pair (S'', S'). Hence there exists a sequence $S'' = S_0, S_1, \ldots, S_m = S'$ with $S_j \in \mathcal{O}_{\vec{k}}$ for all j and such that each transition $S_j \to S_{j+1}$ exchanges at most one element per bin. Appending the initial step $S \to S''$ to this sequence yields the desired path from S to S'.

The above inductive argument yields an explicit polynomial-time procedure to construct the sequence. Given $S, S' \in \mathcal{O}_{\vec{k}}$, form the perturbed weight function $\tilde{v}$ as in the proof. Since $\tilde{v}(S') < \tilde{v}(S)$, the base S is not $\tilde{v}$-optimal, so by Lemma 8 the compressed exchange graph D_S contains a negative directed cycle, which can be found in polynomial time (e.g. by Bellman–Ford; extracting a *simple* negative cycle ensures no bin-vertex v_i is repeated). Exchanging along this cycle produces $S'' \in \mathcal{O}_{\vec{k}}$ that differs from S by at most one element per bin and satisfies $|S' \setminus S''| < |S' \setminus S|$. Repeating for at most $|S' \setminus S| \leq K$ iterations constructs $S = S_0, S_1, \ldots, S_\ell = S'$. ◀

2.5 Deriving an EPTAS

We now show that for every feasible configuration vector $\vec{k}$, there is a Lagrangian–optimal solution that is both feasible for the original linear constraint and nearly optimal in value.

▶ **Lemma 11.** *Fix a configuration vector $\vec{k}$ with $\mathcal{F}_{\vec{k}} \neq \emptyset$, and let $\lambda^\star_{\vec{k}}$, $\mathrm{LR}_{\vec{k}}$, and $\mathcal{O}_{\vec{k}}$ be as defined above. Then, there exists a set $S^\star \in \mathcal{O}_{\vec{k}}$ such that $w(S^\star) \triangleright L$ and the following holds:*

- *if $\mathsf{opt} = \min$, then $v(S^\star) \leq \mathrm{LR}_{\vec{k}} + \Sigma_\Delta(\vec{k})$,*
- *if $\mathsf{opt} = \max$, then $v(S^\star) \geq \mathrm{LR}_{\vec{k}} - \Sigma_\Delta(\vec{k})$.*

Proof. Recall that $\mathcal{O}_{\vec{k}} = \arg \mathsf{opt}_{S \in \mathcal{F}_{\vec{k}}} \phi_{\lambda^\star_{\vec{k}}}(S)$. Choose $S^-, S^+ \in \mathcal{O}_{\vec{k}}$ such that $s_\triangleright(w(S^-) - L)$ is minimized over $\mathcal{O}_{\vec{k}}$ and $s_\triangleright(w(S^+) - L)$ is maximized over $\mathcal{O}_{\vec{k}}$ (These can be computed in polynomial time by lexicographic optimization over $\mathcal{F}_{\vec{k}}$: first optimize $\phi_{\lambda^\star_{\vec{k}}}$, and among ties minimize/maximize $s_\triangleright w(S)$, implemented via weighted matroid intersection).

We claim that $s_\triangleright(w(S^-) - L) \leq 0 \leq s_\triangleright(w(S^+) - L)$. Otherwise, if $s_\triangleright(w(S) - L) > 0$ for all $S \in \mathcal{O}_{\vec{k}}$ (or < 0 for all such S), then perturbing $\lambda^\star_{\vec{k}}$ slightly in the appropriate direction would improve $V_{\vec{k}}(\lambda)$, contradicting the optimality of $\lambda^\star_{\vec{k}}$ in the definition of $\mathrm{LR}_{\vec{k}}$.

For simplicity, we present the remainder of the proof for the minimization case; the maximization case is analogous. Using Eq. (4) with $\mathsf{opt} = \min$ (so $s_{\mathsf{opt}} = -1$), we have $v(S) - \mathrm{LR}_{\vec{k}} = s_\triangleright \lambda^\star_{\vec{k}}\big(w(S) - L\big)$. Since $\lambda^\star_{\vec{k}} \geq 0$, it follows that $s_\triangleright\big(w(S^-) - L\big) \leq 0$ implies $v(S^-) \leq \mathrm{LR}_{\vec{k}}$, and $s_\triangleright\big(w(S^+) - L\big) \geq 0$ implies $v(S^+) \geq \mathrm{LR}_{\vec{k}}$.

By Lemma 9, there is a sequence $S^- = S_0, S_1, \ldots, S_\ell = S^+$ inside $\mathcal{O}_{\vec{k}}$ such that each step exchanges at most one element per bin. Since $v(S_0) \leq \mathrm{LR}_{\vec{k}}$ and $v(S_\ell) \geq \mathrm{LR}_{\vec{k}}$, there exists an index j such that $v(S_j) \leq \mathrm{LR}_{\vec{k}} \leq v(S_{j+1})$.

Combining this inequality with Lemma 7 yields

$$v(S_{j+1}) \leq v(S_j) + \Sigma_\Delta(\vec{k}) \leq \mathrm{LR}_{\vec{k}} + \Sigma_\Delta(\vec{k}).$$

Moreover, since $v(S_{j+1}) \geq \mathrm{LR}_{\vec{k}}$ and $v(S_{j+1}) - \mathrm{LR}_{\vec{k}} = s_\triangleright \lambda^\star_{\vec{k}}\big(w(S_{j+1}) - L\big)$ with $\lambda^\star_{\vec{k}} \geq 0$, it follows that $s_\triangleright\big(w(S_{j+1}) - L\big) \geq 0$, i.e., $w(S_{j+1}) \triangleright L$. Thus $S^\star := S_{j+1}$ is feasible and satisfies the desired bound, which completes the proof.

All objects used above can be computed in polynomial time. First, $\lambda^\star_{\vec{k}}$ is computed as in Lemma 6, and S^-, S^+ are obtained by two lexicographic weighted-matroid-intersection calls over $\mathcal{F}_{\vec{k}}$ (optimize $\phi_{\lambda^\star_{\vec{k}}}$, and among ties minimize/maximize $s_\triangleright w(S)$). Next, Lemma 9 yields an explicit sequence $S^- = S_0, \ldots, S_\ell = S^+$ inside $\mathcal{O}_{\vec{k}}$ with one-per-bin exchanges; scanning this sequence finds an index j with $v(S_j) \leq \mathrm{LR}_{\vec{k}} \leq v(S_{j+1})$, and we output $S^\star := S_{j+1}$. ◀

Algorithm 1 EPTAS for $\mathcal{P}$-MOL .

Input: An instance $\mathcal{A}_{\mathcal{P}} = (E, \mathcal{I}, v, w, L)$, a descriptor $\mathcal{P} = (\mathsf{opt}, \mathcal{F}, \triangleright)$ and $\varepsilon > 0$.
Output: A feasible set S which is $(1 \pm \varepsilon)$-approximation of OPT.

Compute a feasible set $\hat{S}$ and set $\hat{V} := v(\hat{S})$, as guaranteed by Lemma 1.
Define the buckets $B_0, \ldots, B_T \subseteq E$.
$\mathcal{K} \leftarrow \left\{\vec{k} = (k_0, \ldots, k_T) \in \mathbb{Z}_{\geq 0}^{T+1} : \sum_{i=1}^{T} k_i \leq \lceil 2/\varepsilon \rceil,\ \sum_{i=0}^{T} k_i \leq n\right\}$.
$S^{\text{best}} \leftarrow \hat{S}$
for $\vec{k} \in \mathcal{K}$ **do**
    Construct the partition matroid $\mathcal{M}_{\vec{k}}^{P}$ from $\vec{k}$ and define $\mathcal{F}_{\vec{k}} \subseteq 2^E$.
    Compute $\lambda_{\vec{k}}^{\star}$ and $\mathrm{LR}_{\vec{k}}$ (Lemma 6), and compute two sets $S_{\vec{k}}^{-}, S_{\vec{k}}^{+} \in \mathcal{O}_{\vec{k}}$ as in Lemma 11 (with $s_{\triangleright}(w(S_{\vec{k}}^{-}) - L) \leq 0 \leq s_{\triangleright}(w(S_{\vec{k}}^{+}) - L)$).
    Using Lemmas 7, 9 and 11, compute $S_{\vec{k}}^{\star} \in \mathcal{F}_{\vec{k}}$ with $w(S_{\vec{k}}^{\star}) \triangleright L$ and $s_{\mathsf{opt}}\big(\mathrm{LR}_{\vec{k}} - v(S_{\vec{k}}^{\star})\big) \leq \Sigma_{\Delta}(\vec{k})$.
    if $\mathsf{opt} = \min$ and $v(S_{\vec{k}}^{\star}) < v(S^{\text{best}})$ **then**
        $S^{\text{best}} \leftarrow S_{\vec{k}}^{\star}$
    if $\mathsf{opt} = \max$ and $v(S_{\vec{k}}^{\star}) > v(S^{\text{best}})$ **then**
        $S^{\text{best}} \leftarrow S_{\vec{k}}^{\star}$
return S^{best}

Our main result is the following.

▶ **Theorem 12.** *For every fixed $\varepsilon > 0$, there exists an EPTAS for the $\mathcal{P}$-MOL problem. Given an instance $\mathcal{A}_{\mathcal{P}}$, the algorithm runs in time $|E|^{O(1)} \cdot (1/\varepsilon^2)^{O(1/\varepsilon)}$ and outputs a feasible solution S satisfying*

$$\mathit{opt} = \min \implies v(S) \leq (1+\varepsilon)\,\mathit{OPT}, \qquad \mathit{opt} = \max \implies v(S) \geq (1-\varepsilon)\,\mathit{OPT}.$$

Proof. We first describe our EPTAS for the $\mathcal{P}$-MOL class (see Algorithm 1). Using a constant-factor estimate $\hat{V}$ of OPT, we first bucket the elements by their v-values so that elements within the same bucket have nearly equal value. We then enumerate all feasible configuration vectors $\vec{k}$; each such $\vec{k}$ defines a partition matroid $\mathcal{M}_{\vec{k}}^{P}$ and a restricted family $\mathcal{F}_{\vec{k}}$. For every configuration, we solve the Lagrangian relaxation of the linear constraint over $\mathcal{F}_{\vec{k}}$ to obtain $\lambda_{\vec{k}}^{\star}$, $\mathrm{LR}_{\vec{k}}$ and the set $\mathcal{O}_{\vec{k}}$ of Lagrangian-optimal solutions. Finally, using the adjacency property of $\mathcal{O}_{\vec{k}}$ together with a bound on the per-bucket value spread, we find a set $S_{\vec{k}}^{\star} \in \mathcal{F}_{\vec{k}}$ that satisfies the original linear constraint and is within $\Sigma_{\Delta}(\vec{k})$ of the Lagrangian bound. The algorithm returns the best such solution over all configurations.

We now proceed to prove the approximation guarantee of Algorithm 1. Let S^{opt} be an optimal solution and $\vec{k}^{\mathsf{opt}}$ its configuration vector. Lemma 2 establishes that $\vec{k}^{\mathsf{opt}}$ satisfies Eq. (2), so the algorithm considers this vector. In that iteration, we get $\mathsf{OPT}_{\vec{k}^{\mathsf{opt}}} = \mathsf{OPT}$. It holds that

$$\mathsf{opt} = \min \ \Rightarrow\ v(S_{\vec{k}^{\mathsf{opt}}}^{\star}) \leq \mathrm{LR}_{\vec{k}^{\mathsf{opt}}} + \Sigma_{\Delta}(\vec{k}^{\mathsf{opt}}) \leq \mathsf{OPT} + \Sigma_{\Delta}(\vec{k}^{\mathsf{opt}}) \leq \mathsf{OPT} + 8\varepsilon\mathsf{OPT} = (1+8\varepsilon)\mathsf{OPT},$$

$$\mathsf{opt} = \max \ \Rightarrow\ v(S_{\vec{k}^{\mathsf{opt}}}^{\star}) \geq \mathrm{LR}_{\vec{k}^{\mathsf{opt}}} - \Sigma_{\Delta}(\vec{k}^{\mathsf{opt}}) \geq \mathsf{OPT} - \Sigma_{\Delta}(\vec{k}^{\mathsf{opt}}) \geq \mathsf{OPT} - 8\varepsilon\mathsf{OPT} = (1-8\varepsilon)\mathsf{OPT},$$

where the first inequality in each row is by Lemma 11, the second is due to Lemma 5 and the last holds by Lemma 7. Therefore, running the algorithm with parameter $\varepsilon/8$ internally, we obtain the claimed $(1 \pm \varepsilon)$ guarantees. The feasibility of $S_{\vec{k}^{\mathsf{opt}}}^{\star}$ is immediate due to Lemma 11.

The running time is dominated by the enumeration of configuration vectors (Lemma 3) and, for each $\vec{k}$, a polynomial number of calls to (cardinality-constrained) weighted matroid intersection and local exchange operations. This yields a total running time of $|E|^{O(1)} \cdot (1/\varepsilon^2)^{O(1/\varepsilon)}$, as required. ◀

3 Hardness Proof for MIC

▶ **Theorem 13.** *The* Matroid Intersection Cover *problem admits no* EPTAS*, unless* $W[1] = \text{FPT}$.

Proof. We show a reduction from the k-clique problem. Given a graph $G = (V, E)$ and $k \in \mathbb{N}$ we need to decide if there is a clique of size k in G. Let $n = |V|$, and denote by $C(G)$ the set of all cliques in G. We define below a MIC instance, where we assume, without loss of generality, that $0 < k \leq n$ and $n \geq 1$; otherwise, the problem becomes trivial.

Let $V = \{v_1, \ldots, v_n\}$, and for each vertex $v_i \in V$ define a copy u_i, forming the set $U = \{u_1, \ldots, u_n\}$, where $U \cap V = \emptyset$. We refer to u_i and v_i as the two *copies* of the same vertex. Define a function $f : V \cup U \to [n]$ satisfying the following properties:

1. The restriction of f to V is a bijection onto $[n]$.
2. For every $i \in [n]$, it holds that $f(u_i) = f(v_i)$.

For a set $T \subseteq V \cup U$, we write $f(T) := \{f(x) \mid x \in T\}$ for the image of T under f. We also write $\sigma_f(T) := \sum_{x \in T} f(x)$ for the sum of the labels of the elements in T. Using the function f, we define a matroid $M_1 = (V \cup U, \mathcal{F}_1)$, where the independent sets are given by $\mathcal{F}_1 = \mathcal{I}_1 \cup \mathcal{I}_2 \cup \mathcal{I}_3 \cup \mathcal{I}_4$, with the four families defined as follows:

$$\begin{aligned}
\mathcal{I}_1 &= \{T \subseteq V \cup U \;\big|\; |T| < n\} \\
\mathcal{I}_2 &= \{T \subseteq V \cup U \;\big|\; |T| = n, f(T) \neq f(V)\} \\
\mathcal{I}_3 &= \{T \subseteq V \cup U \;\big|\; |T| = n, |T \cap V| \neq k\}. \\
\mathcal{I}_4 &= \{T \subseteq V \cup U \;\big|\; |T| = n, |T \cap V| = k, f(T) = f(V), T \cap V \in C(G)\}.
\end{aligned} \tag{5}$$

▷ Claim 14. M_1 is a matroid.

Proof. We verify the three matroid axioms. First, $\emptyset \in \mathcal{I}_1$ since $0 < n$, and therefore $\emptyset \in \mathcal{F}_1$. Second, for the hereditary property, let $A \in \mathcal{F}_1$ and let $B \subseteq A$. Since $|B| < |A| \leq n$, we have $|B| < n$, hence $B \in \mathcal{I}_1 \subseteq \mathcal{F}_1$. Third, for the exchange property, let $A, B \in \mathcal{F}_1$ such that $|A| > |B|$. We distinguish cases based on the size of B.

- $|B| < n - 1$. Then, for all $e \in A \setminus B$ it holds that $|B + e| < (n - 1) + 1 = n$, and hence $B + e \in \mathcal{I}_1 \subseteq \mathcal{F}_1$. Such an element $e \in A \setminus B$ exists because $|A| > |B|$.
- $|B| = n - 1$ and $|A| = n$. We further distinguish two subcases.
  1. $B \subseteq A$. Then, $A \setminus B = \{e\}$ for some e, and thus $B + e = A \in \mathcal{F}_1$.
  2. $B \not\subseteq A$. Since $|B| = n - 1$ and $|A| = n$, we have $|B \cap A| \leq |B| - 1 = n - 2$. Hence, $|A \setminus B| = |A| - |A \cap B| \geq n - (n - 2) = 2$, so there exist $x, y \in A \setminus B$, such that $x \neq y$. We now consider two cases according to their images under f.
     - **a.** $f(x) \neq f(y)$. Therefore, there is $q \in \{x, y\}$ such that $f(B + q) \neq f(V)$. We conclude that $B + q \in \mathcal{I}_2 \subseteq \mathcal{F}_1$, as required.
     - **b.** $f(x) = f(y)$. By the definition of f, there exists $i \in [n]$ such that $\{x, y\} = \{v_i, u_i\}$ (or vice versa). Hence, adding x or y to B changes the number of selected vertices from V, i.e., $|(B + x) \cap V| \neq |(B + y) \cap V|$. Thus, it follows that for at least one $q \in \{x, y\}$, $|(B + q) \cap V| \neq k$, implying $B + q \in \mathcal{I}_3 \subseteq \mathcal{F}_1$. ◁

We now define a second matroid, $M_2 = (U \cup V, \mathcal{F}_2)$, by

$$\mathcal{F}_2 = \{T \subseteq V \cup U \mid |T \cap \{u_i, v_i\}| \leq 1 \; \forall i \in [n]\}. \tag{6}$$

Note that M_2 is a partition matroid, in which any independent set contains at most one copy of each vertex. We define a weight function $w : (U \cup V) \to \mathbb{R}_{\geq 0}$ by

$$w(x) = \begin{cases} 1 + \frac{f(x)}{2n^2}, & x \in V, \\ \frac{f(x)}{2n^2} & x \in U. \end{cases}$$

and a valuation function $v : (U \cup V) \to \mathbb{R}_{\geq 0}$, given by

$$v(x) = \begin{cases} 1, & x \in V, \\ 0 & x \in U. \end{cases}$$

Finally, we set the threshold $L = k + \frac{\sigma_f(V)}{2n^2}$ and define the MIC instance $\mathcal{A} = (V \cup U, \mathcal{I}, v, w, L)$, where $\mathcal{I} = \mathcal{F}_1 \cap \mathcal{F}_2$.

▷ Claim 15. G contains a clique of size k if and only if $\mathcal{A}$ admits a feasible solution of value k.

Proof. Let Q_V be a clique of size k in G. Define $Q_U = \{u_i \mid i \in [n], v_i \notin Q_V\}$ and $K = Q_V \cup Q_U$. By construction, K contains exactly one copy of each vertex, either v_i or u_i. Therefore, $K \cap V = Q_V \in C(G)$, $|K \cap V| = |Q_V| = k$, and $f(K) = f(V)$, since $f(u_i) = f(v_i)$ for every $i \in [n]$ (and K contains exactly one copy from each vertex, either from U or from V).

Thus, by (5) we have that $K \in \mathcal{I}_4$ implying that $K \in \mathcal{F}_1$. Moreover, since K includes exactly one element from each pair $\{u_i, v_i\}$, it also satisfies the partition constraint and hence $K \in \mathcal{F}_2$. We therefore have $K \in \mathcal{F}_1 \cap \mathcal{F}_2$. In addition, the weight of K satisfies

$$w(K) = w(Q_V) + w(Q_U) = k + \frac{\sigma_f(Q_V)}{2n^2} + \frac{\sigma_f(Q_U)}{2n^2} = k + \frac{\sigma_f(V)}{2n^2} = L,$$

where we use that $|Q_V| = k$, $f(u_i) = f(v_i)$ for all $i \in [n]$, and K contains exactly one copy of each vertex. Hence, K is a feasible solution for $\mathcal{A}$ with value $v(K) = |K \cap V| = |Q_V| = k$.

For the other direction, let Q be a solution for $\mathcal{A}$ with a value k. By the definition of the valuation function v, it follows that $|Q \cap V| = k$. In addition, the total weight of Q is bounded by

$$w(Q) = k + \frac{\sigma_f(Q)}{2n^2} \leq k + \frac{\sigma_f(V)}{2n^2} = L. \tag{7}$$

The first equality follows from $|Q \cap V| = k$, and the inequality holds because $Q \in \mathcal{F}_2$, so Q contains at most one of $\{v_i, u_i\}$ for each $i \in [n]$, so f contributes at most once per vertex of V. Since Q is feasible for $\mathcal{A}$, it must also satisfy $w(Q) \geq L$, and therefore $w(Q) = L$. Combined with (7), this implies $\sigma_f(Q) = \sigma_f(V)$.

Now, notice that if $|Q| < n$, then $\sigma_f(Q) < \sigma_f(V)$, since M_2 ensures that each vertex contributes at most once. Thus we must have $|Q| = n$. We have now established that

$$|Q \cap V| = k, \qquad |Q| = n, \qquad \sigma_f(Q) = \sigma_f(V).$$

Since $Q \in \mathcal{F}_2$, it contains at most one element from each pair $\{u_i, v_i\}$, and therefore the labels in $f(Q)$ are distinct. As $|Q| = n$ and $\sigma_f(Q) = \sigma_f(V) = 1 + \cdots + n$, it follows that $f(Q) = f(V)$. Since Q is feasible, it belongs to one of the families defining $\mathcal{F}_1$; the only possible family is $\mathcal{I}_4$. By (5), we conclude that $Q \cap V \in C(G)$. Hence, $Q \cap V$ is a clique of size k in G. ◁

To complete the proof, assume toward a contradiction that there exists an EPTAS for MIC. Using this scheme, we can decide whether G contains a clique of size k in FPT time. Consider the instance $\mathcal{A}$ constructed above, and run the EPTAS with $\varepsilon = \frac{1}{2k}$. Observe that any feasible solution for $\mathcal{A}$ must include at least k vertices from V. The weight constraint encoded by w and L enforces this requirement, as each vertex from V contributes 1 to the value function v.

By Claim 15, G has a clique of size k if and only if $\mathcal{A}$ admits a feasible solution of value exactly k. If such a solution for $\mathcal{A}$ exists, then the EPTAS would yield a solution whose value is at most $(1+\varepsilon)k = k + \frac{1}{2} < k+1$. Since the valuation function v is integral, the only integer value below $k+1$ is k, and therefore the scheme must return a solution of value exactly k. Thus, the EPTAS correctly determines whether $\mathcal{A}$ has a solution of value k and, by Claim 15, whether G contains a clique of size k.

The running time of the EPTAS is $g(\frac{1}{\varepsilon}) \cdot m^{O(1)}$ for some computable function g, where m is the encoding size of the k-clique instance. Since $\varepsilon = \Theta(\frac{1}{k})$, our reduction decides the k-clique problem in time $g(k) \cdot m^{O(1)}$. As the k-clique problem is known to be W[1]-Hard [18], it implies that there is no EPTAS for MIC unless $W[1] = \text{FPT}$. ◀

4 Discussion

In this paper, we present tight approximation results for a broad class of matroid optimization problems with a linear constraint, along with a hardness result for MIC. Our unified EPTAS for all $\mathcal{P}$-MOL problems resolves the complexity status of this class, and the hardness of MIC suggests that such a unified EPTAS is unlikely to extend to *all* $\mathcal{P}$-MOL variants once the single matroid constraint is replaced by a *matroid intersection* constraint.

A natural direction for future work is to characterize the approximability landscape of $\mathcal{P}$-MOL variants under matroid intersection constraints. In particular, the existence of a PTAS for MIC remains open.

Finally, it would be interesting to identify settings in which stronger guarantees are attainable. For instance, for graphic matroids and for linear matroids given by an explicit representation, we obtain an EPTAS, while the existence of an FPTAS in these cases remains open.

References

1 André Berger, Vincenzo Bonifaci, Fabrizio Grandoni, and Guido Schäfer. Budgeted matching and budgeted matroid intersection via the gasoline puzzle. *Mathematical Programming*, 128(1):355–372, 2011. doi:10.1007/S10107-009-0307-4.

2 Carl Brezovec, Gerard Cornuéjols, and Fred Glover. Two algorithms for weighted matroid intersection. *Mathematical Programming*, 36(1):39–53, 1986. doi:10.1007/BF02591988.

3 Carl Brezovec, Gérard Cornuéjols, and Fred Glover. A matroid algorithm and its application to the efficient solution of two optimization problems on graphs. *Mathematical Programming*, 42(1):471–487, 1988. doi:10.1007/BF01589417.

4 Karl Bringmann and Vasileios Nakos. A fine-grained perspective on approximating subset sum and partition. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1797–1815. SIAM, 2021. doi:10.1137/1.9781611976465.108.

5 Alberto Caprara, Hans Kellerer, Ulrich Pferschy, and David Pisinger. Approximation algorithms for knapsack problems with cardinality constraints. *European Journal of Operational Research*, 123(2):333–345, 2000. doi:10.1016/S0377-2217(99)00261-1.

6 Venkatesan T. Chakaravarthy, Anamitra Roy Choudhury, Sivaramakrishnan R. Natarajan, and Sambuddha Roy. Knapsack Cover Subject to a Matroid Constraint. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2013)*, volume 24 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 275–286. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2013. doi:10.4230/LIPIcs.FSTTCS.2013.275.

7 Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Multi-budgeted matchings and matroid intersection via dependent rounding. In *Proc. SODA*, pages 1080–1097, 2011. doi:10.1137/1.9781611973082.82.

8 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 2022.

9 Mingyang Deng, Ce Jin, and Xiao Mao. Approximating knapsack and partition via dense subset sums. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2961–2979. SIAM, 2023. doi:10.1137/1.9781611977554.CH113.

10 Ilan Doron-Arad, Fabrizio Grandoni, and Ariel Kulik. Unsplittable Flow on a Short Path. In *19th International Symposium on Parameterized and Exact Computation (IPEC 2024)*, 2024.

11 Ilan Doron-Arad, Ariel Kulik, and Hadas Shachnai. Budgeted matroid maximization: a parameterized viewpoint. *IPEC*, 2023.

12 Ilan Doron-Arad, Ariel Kulik, and Hadas Shachnai. An EPTAS for budgeted matching and budgeted matroid intersection via representative sets. In *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023)*, pages 49:1–49:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ICALP.2023.49.

13 Ilan Doron-Arad, Ariel Kulik, and Hadas Shachnai. An EPTAS for budgeted matroid independent set. In *Proc. SOSA*, pages 69–83, 2023. doi:10.1137/1.9781611977585.CH7.

14 Ilan Doron-Arad, Ariel Kulik, and Hadas Shachnai. An FPTAS for budgeted laminar matroid independent set. *Operations Research Letters*, 51(6):632–637, 2023. doi:10.1016/J.ORL.2023.10.005.

15 Ilan Doron-Arad, Ariel Kulik, and Hadas Shachnai. Lower bounds for matroid optimization problems with a linear constraint. In *51st International Colloquium on Automata, Languages, and Programming (ICALP), July 8–12, 2024*, pages 56:1–56:20, 2024. doi:10.4230/LIPIcs.ICALP.2024.56.

16 Ilan Doron-Arad, Ariel Kulik, and Hadas Shachnai. You (almost) can't beat brute force for 3-matroid intersection. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 5151–5171. SIAM, 2026. doi:10.1137/1.9781611978971.186.

17 Ilan Doron-Arad, Hadas Shachnai, Gilad Shmerler, and Inbal Talgam-Cohen. An algorithm-to-contract framework without demand queries. *arXiv preprint arXiv:2507.20038*, 2025. doi:10.48550/arXiv.2507.20038.

18 Rod G Downey and Michael R Fellows. Fixed-parameter tractability and completeness II: On completeness for W[1]. *Theoretical Computer Science*, 141(1-2):109–131, 1995. doi:10.1016/0304-3975(94)00097-3.

19 Fabrizio Grandoni and Rico Zenklusen. Approximation schemes for multi-budgeted independence systems. In *Proc. ESA*, pages 536–548, 2010. doi:10.1007/978-3-642-15775-2_46.

20 Refael Hassin and Asaf Levin. An efficient polynomial time approximation scheme for the constrained minimum spanning tree problem using matroid intersection. *SIAM Journal on Computing*, 33(2):261–268, 2004. doi:10.1137/S0097539703426775.

21 Sung-Pil Hong, Sung-Jin Chung, and Bum Hwan Park. A fully polynomial bicriteria approximation scheme for the constrained spanning tree problem. *Operations Research Letters*, 32(3):233–239, 2004. doi:10.1016/J.ORL.2003.06.003.

22 Hans Kellerer, Ulrich Pferschy, and David Pisinger. The multiple-choice knapsack problem. In *Knapsack Problems*, pages 317–347. Springer, 2004.

23 Ariel Kulik and Hadas Shachnai. There is no EPTAS for two-dimensional knapsack. *Information Processing Letters*, 110(16):707–710, 2010. doi:10.1016/J.IPL.2010.05.031.

24 Eugene L. Lawler. Fast approximation algorithms for knapsack problems. *Mathematics of Operations Research*, 4(4):339–356, 1979. doi:10.1287/MOOR.4.4.339.

25 Nimrod Megiddo. Combinatorial optimization with rational objective functions. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing*, pages 1–12, 1978. doi:10.1145/800133.804326.

26 Ram Ravi and Michel X Goemans. The constrained minimum spanning tree problem. In *Scandinavian Workshop on Algorithm Theory*, pages 66–75. Springer, 1996.

27 Yang Yang, Mikhail Y Kovalyov, and Yong Zhang. An improved fptas for budgeted laminar matroid independent set. *Operations Research Letters*, page 107307, 2025. doi:10.1016/J.ORL.2025.107307.

Strong Inapproximability for a Promise Rank Problem

Venkatesan Guruswami
Simons Institute for the Theory of Computing, University of California, Berkeley, CA, USA
Departments of EECS and Mathematics, University of California, Berkeley, CA, USA

Xuandi Ren
Department of EECS, University of California, Berkeley, CA, USA

Shaoxuan Tang
Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China

Abstract

Given a linear subspace of $n \times n$ matrices over $\mathbb{F}_{2^r}$ that is promised to contain a matrix of rank 1, we prove that it is hard to find a matrix of rank $n^{o(1/\log\log n)}$, assuming NP doesn't have sub-exponential algorithms. In addition to being a basic problem, the hardness of this problem, even for the exact version, drove recent PCP-free inapproximability results for minimum distance and shortest vector problems concerning codes and lattices.

The proof combines the concept of superposition soundness introduced by Khot and Saket with moment matrices. To produce a rank-gap of 1 vs. k, the reduction runs in time $n^{O(\log k)}$. We also give another moment-matrix-based construction which runs in time $n^{O(k)}$ but works for any finite field $\mathbb{F}_q$.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness; Theory of computation → Approximation algorithms analysis

Keywords and phrases rank minimization, inapproximability, promise problems, PCP, moment matrices

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.19

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2605.11545

Funding *Venkatesan Guruswami*: Research supported in part by NSF grant CCF-2211972, a DARPA grant under Contract No. HR0011262E031, and a Simons Investigator award.
Xuandi Ren: Supported in part by NSF grant CCF-2228287 and V.G.'s Simons Investigator award.

Acknowledgements Some of this work was done while Shaoxuan Tang was visiting UC Berkeley. The assistance of ChatGPT was used with some of the proofs in Section 4. All proofs were carefully verified by the authors.

1 Introduction

Given a linear subspace $\mathcal{L} \subseteq \mathbb{F}^{N\times N}$ that is promised to contain a matrix of rank one, we study the problem of finding a nonzero matrix in $\mathcal{L}$ of minimum rank.

This problem is closely related to the problem of finding the *minimum rank distance* of a rank-metric code [5, 9, 17]. In fact, there is a simple and classical embedding of Hamming-metric codes into this setting. Given a linear code $C \subseteq \mathbb{F}^N$, consider the diagonal matrix code

$$\mathrm{Diag}(C) := \{\mathrm{diag}(c) : c \in C\} \subseteq \mathbb{F}^{N\times N}.$$


© Venkatesan Guruswami, Xuandi Ren, and Shaoxuan Tang;
licensed under Creative Commons License CC-BY 4.0



For every $c \in C$,

$$\mathrm{rank}(\mathrm{diag}(c)) = \|c\|_0,$$

and hence the minimum rank distance of $\mathrm{Diag}(C)$ is exactly the minimum Hamming distance of C. Consequently, hardness results for the Minimum Distance Problem (MDP) [7, 4, 1, 15, 3] directly translate to hardness for approximating the minimum rank in a matrix subspace.

However, this diagonal reduction also explains what it does *not* prove. The YES case inherited from MDP only promises a nonzero matrix of rank $d(C)$, the minimum distance of the code, which is generally not 1. In fact, it is trivial to check if the minimum distance of a code is 1. It is therefore natural to ask whether the problem remains hard under the stronger promise that the subspace contains a nonzero matrix of smallest possible rank, namely rank 1.

A second motivation comes from recent PCP-free inapproximability results for sparse vector problems, including the Minimum Distance Problem and the Shortest Vector Problem [3]. These reductions start from systems of quadratic equations and view each quadratic constraint as a linear constraint on a matrix: if $X = xx^\top$, then a quadratic form in x becomes a linear form in the entries of X. Hence honest solutions to the original quadratic system correspond to rank-one matrices. The main soundness issue is to rule out spurious higher-rank matrices.

In [3], this issue is handled via a non-overlap lemma. Specifically, if C is a linear code of distance $d(C)$ over $\mathbb{F}$, then honest rank-one matrices in the tensor code $C \otimes C$ attain the minimum Hamming weight $d(C)^2$. On the other hand, every matrix in $C \otimes C$ of rank at least 2 has Hamming weight at least $\alpha \cdot d(C)^2$, for some $\alpha > 1$ depending on $\mathbb{F}$. Thus the Hamming-weight gap driving the MDP hardness is, at its core, a structural gap between rank-one matrices and matrices of rank at least two. This suggests stripping away the Hamming-weight objective and asking for the rank gap directly.

In this work, we show strong hardness of distinguishing

$$\exists\, 0 \neq A \in \mathcal{L} \text{ with } \mathrm{rank}(A) = 1 \qquad \text{from} \qquad \forall\, 0 \neq A \in \mathcal{L},\ \mathrm{rank}(A) > k$$

over any fixed finite field of characteristic 2. Specifically, we give a deterministic reduction from 3SAT producing a rank gap of 1 versus k, with matrix dimension $n^{O(\log k)}$. The resulting inapproximability consequences are summarized in the following main theorem.

▶ **Theorem 1.1.** *Let*

$$\mathrm{minrank}(\mathcal{L}) := \min\{\mathrm{rank}(A) : 0 \neq A \in \mathcal{L}\}.$$

For every fixed integer $r \geq 1$, no polynomial-time algorithm can, given a linear subspace $\mathcal{L} \subseteq \mathbb{F}_{2^r}^{N \times N}$, distinguish between

- *(YES)* $\mathrm{minrank}(\mathcal{L}) = 1$,
- *(NO)* $\mathrm{minrank}(\mathcal{L}) > \gamma$,

in the following regimes:

(a) *assuming* $\mathsf{NP} \neq \mathsf{P}$, *when $\gamma > 1$ is any constant;*

(b) *assuming* $\mathsf{NP} \not\subseteq \mathsf{DTIME}(2^{\log^{O(1)} n})$, *when* $\gamma = 2^{(\log N)^{1-\epsilon}}$ *for any fixed* $0 < \epsilon < 1$;

(c) *assuming* $\mathsf{NP} \not\subseteq \bigcap_{\delta > 0} \mathsf{DTIME}(2^{n^\delta})$, *when* $\gamma = N^{c/\log\log N}$ *for some fixed constant* $c > 0$.

We present the reduction in Section 3. The proof combines the superposition-soundness framework of Khot and Saket [12] with linearized moment matrices. A satisfying assignment to the starting quadratic system gives a rank-one moment matrix. Conversely, a low-rank

feasible matrix can be decomposed into a bounded number of symmetric rank-one pieces; these pieces behave like several assignments satisfying the quadratic system in superposition. The soundness analysis of Khot–Saket soundness rules out such superpositions, while the moment-matrix equal-union constraints ensure that the remaining zero-sum case cannot hide a nonzero low-rank matrix.

We also include a simpler direct moment-matrix reduction in Section 4. This construction avoids the superposition-soundness and works over every finite field $\mathbb{F}_q$, at the cost of producing matrices of dimension $n^{O(k)}$. The corresponding inapproximability statement is the following.

▶ **Theorem 1.2.** *Let*

$$\operatorname{minrank}(\mathcal{L}) := \min\{\operatorname{rank}(A) : 0 \neq A \in \mathcal{L}\}.$$

For every fixed finite field $\mathbb{F}_q$, no polynomial-time algorithm can, given a linear subspace $\mathcal{L} \subseteq \mathbb{F}_q^{N\times N}$, distinguish between

- *(YES)* $\operatorname{minrank}(\mathcal{L}) = 1$,
- *(NO)* $\operatorname{minrank}(\mathcal{L}) > \gamma$,

in the following regimes:

(a) *assuming* NP $\neq$ P, *when $\gamma > 1$ is any constant;*

(b) *assuming* NP $\not\subseteq$ DTIME$(2^{\log^{O(1)} n})$, *when $\gamma = (\log N)^{1-\epsilon}$ for any fixed $0 < \epsilon < 1$;*

(c) *assuming* NP $\not\subseteq \bigcap_{\delta>0}$ DTIME(2^{n^δ}), *when $\gamma = c \log N / \log\log N$ for some fixed constant $c > 0$.*

Proof overview

The key ingredient in both reductions is the pseudo-moment matrix. Fix a moment level d. We introduce a pseudo-moment coordinate y_R for every Boolean monomial $x^R := \prod_{i\in R} x_i$ of degree at most $2d$. Note that the vector (y_R) is not assumed to come from an actual Boolean assignment, for which reason it is called pseudo-moment vector. The associated degree-d pseudo-moment matrix[1] is defined by

$$H_d(y)_{S,T} = y_{S\cup T}. \qquad (|S|, |T| \leq d)$$

In the matrix formulation, we impose the equal-union constraints

$$A_{S,T} = A_{S',T'} \qquad \text{whenever} \qquad S \cup T = S' \cup T'.$$

This way, the matrix is not an arbitrary linearization of the products $x^S x^T$. All factorizations of the same Boolean monomial $x^{S\cup T}$ are forced to share a single pseudo-moment coordinate.

This hidden redundancy is what makes rank a useful test. If a pseudo-moment y_R is nonzero, it is not isolated in one entry of the matrix: it appears in every entry whose row and column labels union to R. Therefore a feasible low-rank matrix is constrained not only by the original equations, but also by the truncated Boolean monomial algebra encoded by these equal-union identities.

[1] This is the standard moment-matrix object from the Sum-of-Squares literature: for a pseudoexpectation $\widetilde{\mathbb{E}}$ over the Boolean cube, the moment matrix has entries $\widetilde{\mathbb{E}}[x^S x^T] = \widetilde{\mathbb{E}}[x^{S\cup T}]$. The Lasserre/SoS hierarchy additionally imposes normalization and positive-semidefiniteness of moment matrix. See, e.g., [13, 16, 14, 2].

We first record the simple but powerful observation related to the rank of pseudo-moment matrices that is used in both reductions, and then give an overview of the two reductions separately.

▶ **Lemma 1.3** (Informal structural lemma, used in Lemma 3.10 and Lemma 4.3)**.** *Let A be a matrix satisfying the above equal-union constraints, and let y be the pseudo-moment vector it represents. Suppose R is a minimum-size set with $y_R \neq 0$. Then,*

$$\operatorname{rank}(A) \geq \binom{|R|}{\lfloor |R|/2 \rfloor}.$$

The proof idea is to look at the submatrix with rows indexed by the $\lfloor |R|/2 \rfloor$-subsets $F \subseteq R$ and columns indexed by the $\lceil |R|/2 \rceil$-subsets $G \subseteq R$. Its (F, G) entry is $y_{F \cup G}$. By the minimality of R, this entry is zero unless $F \cup G = R$. Hence the submatrix is a permutation matrix, giving the rank lower bound.

The two reductions use this observation in different ways. In the $n^{O(\log k)}$ superposition reduction, the observation turns the zero-sum case from Khot–Saket soundness into the conclusion that the whole low-rank matrix is zero. In the $n^{O(k)}$ direct pseudo-moment reduction, the same observation forces the existence of a nonzero flat level from which one can round to an honest Boolean solution.

The $n^{O(\log k)}$ superposition reduction over $\mathbb{F}_{2^r}$

We first prove the rank gap over $\mathbb{F}_2$ and then extend it to $\mathbb{F}_{2^r}$ in a black-box way. We adapt the Khot–Saket superposition reduction [12] to produce a constant-free quadratic system $\mathcal{Q}$. After introducing a homogenizing variable x_0, and introducing variables y_S for non-constant monomials x^S of degree at most $d = \Theta(\log k)$, we obtain a quadratic system with the following property. In the YES case, an honest satisfying assignment gives a solution to $\mathcal{Q}$. In the NO case, if t assignments satisfy $\mathcal{Q}$ in superposition, meaning that the sum of their evaluations on every equation of $\mathcal{Q}$ is zero, then their coordinate-wise sum is the zero assignment.

After that, for each quadratic equation

$$\sum_{S,T} c_{S,T} y_S y_T + \sum_R b_R y_R = 0$$

in $\mathcal{Q}$, we linearize it as

$$\sum_{S,T} c_{S,T} A_{S,T} + \sum_R b_R A_{R,R} = 0,$$

and we also impose all equal-union constraints.

For completeness, an honest satisfying assignment gives a feasible rank-one moment matrix.

For soundness, suppose in the NO case that a nonzero feasible matrix A has rank at most k. A decomposition lemma[2] shows

$$A = \sum_{i=1}^{t} u^{(i)} u^{(i)\top}, \qquad t \leq \left\lfloor \frac{3k}{2} \right\rfloor.$$

[2] This decomposition is not obvious: we require a sum of *symmetric* rank-1 terms, which is more restrictive than a general rank decomposition.

Substituting this decomposition into the linearized constraints shows that the assignments $u^{(i)}$ satisfy $\mathcal{Q}$ in superposition[3]. By the soundness guarantee, the sum of these assignments is zero. Consequently, every diagonal entry of A vanishes:

$$A_{S,S} = \sum_i (u_S^{(i)})^2 = \sum_i u_S^{(i)} = 0.$$

The equal-union constraints then imply $A_{S,T} = A_{S\cup T, S\cup T} = 0$ whenever $|S \cup T| \le d$.

We still need to rule out nonzero entries whose union has size larger than d. Suppose such an entry exists, and choose $A_{S,T} \neq 0$ with $|S \cup T|$ minimum. Put $R = S \cup T$. By the structural lemma,

$$\operatorname{rank}(A) \ge \binom{|R|}{\lfloor |R|/2 \rfloor} \ge \binom{d+1}{\lfloor (d+1)/2 \rfloor} > k,$$

contradicting the rank assumption.

Finally, to work over $\mathbb{F}_{2^r}$, we interpret the same $\mathbb{F}_2$-linear equations over the extension field; a nonzero rank-at-most-k matrix over $\mathbb{F}_{2^r}$ would descend, via an $\mathbb{F}_2$-linear functional, to a nonzero matrix over $\mathbb{F}_2$ of rank at most rk. So we just run the base-field reduction with gap rank parameter rk.

The $n^{O(k)}$ direct reduction. This construction starts from QuadEq over an arbitrary finite field $\mathbb{F}_q$. We set $d = k$ and build the pseudo-moment matrix whose rows and columns are indexed by all monomials of degree at most d, including the empty monomial. We impose the same equal-union constraints, and for each equation of the QuadEq instance

$$f_\ell(x) = \sum_{U \subseteq [n],\ |U| \le 2} c_{\ell,U} x^U$$

we impose the following pseudo-moment versions of the identities $x^W f_\ell(x) = 0$:

$$\sum_{\substack{U \subseteq [n] \\ |U| \le 2}} c_{\ell,U} y_{U \cup W} = 0 \qquad \forall W,\ |W| \le 2d - 2\ .$$

These are the pseudo-moment versions of the identities $x^W f_\ell(x) = 0$. The completeness is immediate: a Boolean solution gives a rank-one moment matrix solution, and it is nonzero because the empty-coordinate entry is 1.

For soundness, suppose the source instance is unsatisfiable but there is a nonzero feasible pseudo-moment matrix $H_d(y)$ of rank at most d. Let $r_e = \operatorname{rank} H_e(y)$; this sequence is nondecreasing in e. The structural lemma shows that the first nonzero level already has a reasonably large rank that the sequence r_e cannot keep increasing strictly up to level d. Hence there must be a nonzero flat level

$$r_e = r_{e+1} > 0$$

for some $e < d$.

[3] The constant-free assumption is needed here. If a quadratic equation has a nonzero constant term, substituting a decomposition $A = \sum_{i=1}^t u^{(i)} u^{(i)\top}$ counts that constant with a parity depending on the number of summands, so a linearized feasible matrix yields the desired superposition condition only when t is odd. Huang [11], building on the Khot–Saket superposition framework, uses precisely this oddness phenomenon: he proves that superposition satisfaction by an odd number of assignments is equivalent to odd-covering.

At such a flat level, multiplying a column label by x_i creates no new column direction. Thus multiplication by x_i defines a linear operator T_i on the column space C_e of $H_e(y)$. These operators satisfy the Boolean rules

$$T_i^2 = T_i, \qquad T_iT_j = T_jT_i,$$

and the localizing constraints translate into the operator identities $f_\ell(T) = 0$ on C_e for every source equation f_ℓ. Since the T_i are commuting projections over $\mathbb{F}_q$, they have a common eigenvector with eigenvalues $a_i \in \{0,1\}$. The tuple $a = (a_1, \ldots, a_n)$ is therefore a Boolean point, and the identities $f_\ell(T) = 0$ imply $f_\ell(a) = 0$ for all ℓ, contradicting unsatisfiability.

2 Preliminaries

We now formally define the concepts related to monomial-indexing and pseudo-moments used in Section 3 and Section 4, which we already sketched in the introduction.

Let $[n] := \{1, 2, \ldots, n\}$ for any positive integer n. For a subset S, we write $x^S := \prod_{i \in S} x_i$, and we set $x^\emptyset := 1$. All products of monomials and polynomials are taken in the Boolean squarefree monomial algebra. Equivalently, over the relevant field we identify polynomials modulo the relations $x_i^2 = x_i$, so that $x^S x^T = x^{S \cup T}$.

Write

$$\mathcal{U}_{n,d} := \{S \subseteq \{0, 1, \ldots, n\} : 1 \le |S| \le d\};$$
$$\mathcal{V}_{n,d} := \{S \subseteq \{1, \ldots, n\} : 0 \le |S| \le d\}.$$

Here $\mathcal{U}_{n,d}$ is used for the homogenized $n^{O(\log k)}$ construction: it allows the coordinate 0 but excludes the empty monomial. The set $\mathcal{V}_{n,d}$ is used for the direct $n^{O(k)}$ construction: it uses only the original variables and includes the empty monomial.

When $\mathcal{W}_d$ is either $\mathcal{U}_{n,d}$ or $\mathcal{V}_{n,d}$, define a *monomial assignment of degree d* to be a map

$$\sigma : \{x^S : S \in \mathcal{W}_d\} \to \mathbb{F}.$$

Equivalently, any monomial assignment of degree d can be written as a *degree-d pseudo-moment assignment*, or *pseudo-moment vector*,

$$y = (y_R)_{R \in \mathcal{W}_d}.$$

When a degree-$2d$ pseudo-moment vector is given, its associated *pseudo-moment matrix* is defined as

$$H_d(y) := \left(y_{S \cup T}\right)_{S,T \in \mathcal{W}_d}.$$

If a is an actual Boolean point in $\{0,1\}^{n+1}$ (or $\{0,1\}^n$, respectively), the "honest" moment assignment generated by a is $y_R = a^R$. The corresponding moment vector is

$$v_d(a) := (a^S)_{S \in \mathcal{W}_d},$$

and in this case

$$H_d(y) = v_d(a) v_d(a)^\top.$$

In particular, $H_d(y)$ has rank one whenever $v_d(a) \neq 0$.

On the other hand, any matrix $A \in \mathbb{F}^{\mathcal{W}_d \times \mathcal{W}_d}$ satisfying

$$A_{S,T} = A_{S',T'} \qquad \text{whenever } S \cup T = S' \cup T',$$

is a pseudo-moment matrix $H_d(y)$ for some unique pseudo-moment vector $y \in \mathbb{F}^{\mathcal{W}_{2d}}$. We therefore sometimes state the linear constraints on the pseudo-moment matrix in terms of the associated pseudo-moment vector y.

3 An $n^{O(\log k)}$ Construction over $\mathbb{F}_{2^r}$

In this section, we prove the following theorem.

▶ **Theorem 3.1.** *For every fixed integer $r \geq 1$ and every integer $k \geq 1$, there is a deterministic $n^{O(\log k)}$-time reduction which maps a* 3SAT *instance on n variables to a linear subspace*

$$\mathcal{L} \subseteq \mathbb{F}_{2^r}^{N \times N}, \qquad N = n^{O(\log k)},$$

such that:

- *(YES) if the input formula is satisfiable, then $\mathcal{L}$ contains a nonzero matrix of rank 1;*
- *(NO) if the input formula is unsatisfiable, then $\mathcal{L}$ contains no nonzero matrix of rank at most k.*

To prove Theorem 3.1, we first prove the result over $\mathbb{F}_2$, then transfer it to any $\mathbb{F}_{2^r}$ by a rank-descent argument. Our reduction has four steps:

1. First, we reduce 3SAT to a degree-d polynomial equation system $\mathcal{B}$ over $\mathbb{F}_2$ on the variables $x_0, x_1, \ldots, x_n$. The homogenizing variable x_0 replaces the constant 1 and makes every equation in $\mathcal{B}$ have zero constant term.
2. Next, we replace each nonconstant monomial x^S, $S \in \mathcal{U}_{n,d}$, by a new variable y_S, and we add the moment constraints $y_S y_T = y_{S \cup T}$. This produces a quadratic equation system $\mathcal{Q}$ with zero constant term. Furthermore, we obtain Khot–Saket superposition soundness: in the NO case, if t assignments $\tau^{(1)}, \ldots, \tau^{(t)}$ satisfy every equation of $\mathcal{Q}$ in superposition (see Definition 3.5), then the sum of the t assignments

$$\tau := \sum_{i=1}^{t} \tau^{(i)}$$

must vanish on every y-coordinate.
3. We then linearize the quadratic system $\mathcal{Q}$ by introducing matrix variables $A_{S,T}$ for the products $y_S y_T$. By adding equal-union constraints to enforce the Boolean monomial identity $x^S x^T = x^{S \cup T}$, and by the rank observation in Lemma 1.3, we get the rank gap over $\mathbb{F}_2$.
4. Finally, we extend the hardness from $\mathbb{F}_2$ to $\mathbb{F}_{2^r}$ by interpreting the same homogeneous linear constraints over the extension field. The rank-descent lemma Lemma 3.12 converts any nonzero matrix of rank at most k over $\mathbb{F}_{2^r}$ into a nonzero matrix over $\mathbb{F}_2$ of rank at most rk. Running the $\mathbb{F}_2$ construction with rank parameter rk gives the claimed gap over $\mathbb{F}_{2^r}$.

The four steps are carried out in Sections 3.1–3.4, respectively. For orientation, the first three variable-changing steps are summarized in Figure 1; the corresponding variables and their roles are listed in Table 1.

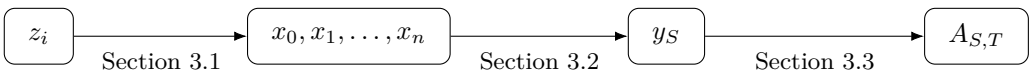


Figure 1 Roadmap of the reduction steps in this section.

Table 1 Variables defined in this section and their roles.

Variables	Meaning	Used in
z_i	original 3SAT variables	starting NP-hardness source
$x_0, x_1, \ldots, x_n$	homogenized Boolean variables	degree-d polynomial system $\mathcal{B}$ with no constant term; x_0 replaces 1
y_S	variable for monomial x^S	quadratic system $\mathcal{Q}$ with Khot–Saket superposition soundness
$A_{S,T}$	matrix variable linearizing $y_S y_T$	linear matrix subspace $\mathcal{L}_d(\mathcal{Q})$ for the rank-gap reduction

3.1 Reducing from 3Sat to constant-free polynomial equations

In this subsection, before presenting our main reduction in Theorem 3.4, we first introduce the following duality fact used in [12]: any assignment to all nonconstant monomials of degree at most d can be represented as the sum of evaluations at actual points of $\mathbb{F}_2^{n+1}$.

▶ **Lemma 3.2** (Monomial assignments as sums of points, [12]). *Let $\mathbb{F}_2[x]_{\le d}$ denote the $\mathbb{F}_2$-vector space of multilinear polynomials in variables $x_0, \ldots, x_n$ of degree at most d. For every monomial assignment*

$$\sigma : \{x^S : S \in \mathcal{U}_{n,d}\} \to \mathbb{F}_2,$$

define $\sigma(1) = 1$ and extend σ by linearity to all polynomials in $\mathbb{F}_2[x]_{\le d}$. Then, there is a subset $\beta \subseteq \mathbb{F}_2^{n+1}$ such that

$$\sigma(q) = \sum_{a \in \beta} q(a)$$

holds for every $q \in \mathbb{F}_2[x]_{\le d}$.

Since $\sigma(1) = 1$, any set β representing σ must have odd cardinality. Among all sets representing σ in this sense, fix one with minimum cardinality and denote it by β_σ. Due to the page limit, we defer the proofs of Lemma 3.2 and the following point-isolator lemma, Lemma 3.3, to the full version of this paper.

▶ **Lemma 3.3** (Point isolator). *Let $\rho \ge 0$ be an integer, $T \subseteq \mathbb{F}_2^{n+1}$ with $|T| < 2^\rho$, and let $a \in T$. Then there exists a multilinear polynomial q of degree at most ρ such that*

$$q(a) = 1 \qquad \text{and} \qquad q(b) = 0 \quad \text{for all } b \in T \setminus \{a\}.$$

We now give the reduction from 3SAT to a system of degree-d constant-free polynomial equations. This is a constant-free analogue of the Khot–Saket construction. We homogenize the usual Boolean constraints by introducing a variable x_0, which plays the role of the constant 1 in honest assignments, and by adding the equations $x_i(x_i + x_0) = 0$. If $x_0 = 1$, the clause polynomials encode the original 3SAT instance. If $x_0 = 0$, these equations force every variable to vanish as well.

▶ **Theorem 3.4** (Constant-free low-weight soundness). *For every integer $d \ge 4$, there is a deterministic $n^{O(d)}$-time reduction from a 3SAT instance on n variables to a system $\mathcal{B}$ of constant-free degree-d polynomial equations in $n+1$ variables over $\mathbb{F}_2$ such that:*

- *(YES) if the input formula is satisfiable, then $\mathcal{B}$ has a satisfying assignment $a \in \mathbb{F}_2^{n+1}$ with $a_0 = 1$;*
- *(NO) if the input formula is unsatisfiable and σ is a monomial assignment of degree d satisfying all equations in $\mathcal{B}$, then either $\beta_\sigma = \{\mathbf{0}\}$ or $|\beta_\sigma| \geq 2^{d-3}$.*

Proof. Let the input formula be φ with variables $z_1, \ldots, z_n$ and clauses $C_1, \ldots, C_m$. We introduce variables $x_0, x_1, \ldots, x_n$. For a literal ℓ, define its false-literal form by

$$h_\ell(x) = \begin{cases} x_0 + x_i & \text{if } \ell = z_i, \\ x_i & \text{if } \ell = \neg z_i. \end{cases}$$

When $x_0 = 1$, this is 1 if and only if the literal ℓ is false. For a clause $C_j = \ell_{j,1} \vee \ell_{j,2} \vee \ell_{j,3}$, define the clause polynomial

$$p_j(x) := h_{\ell_{j,1}}(x) h_{\ell_{j,2}}(x) h_{\ell_{j,3}}(x).$$

For every original variable, define the Booleanity polynomial

$$b_i(x) := x_i(x_i + x_0).$$

Every p_j and b_i has no constant term, and their degrees are at most 3 and 2, respectively.

The system $\mathcal{B}$ consists of the following equations:

(1) $x^S p_j(x) = 0$ for every $j \in [m]$ and every monomial x^S, including $S = \emptyset$, of degree at most $d - 3$;

(2) $x^S b_i(x) = 0$ for every $i \in [n]$ and every monomial x^S, including $S = \emptyset$, of degree at most $d - 2$.

All equations have degree at most d and no constant term.

In the YES case, take a satisfying Boolean assignment $(a_1, \ldots, a_n)$ to φ and set $a_0 = 1$. Then each $b_i(a)$ vanishes. In every clause at least one literal is true, so at least one of the three corresponding false-literal forms is zero, and hence every $p_j(a)$ vanishes. All monomial multiples in $\mathcal{B}$ therefore vanish at a as well.

Now assume φ is unsatisfiable, and let σ be a monomial assignment satisfying all equations in $\mathcal{B}$. Suppose, for contradiction, that $\beta_\sigma \neq \{\mathbf{0}\}$ and $|\beta_\sigma| < 2^{d-3}$. By the parity observation above, β_σ is nonempty; since it is not $\{\mathbf{0}\}$, it contains a nonzero point. Choose such a point $a \in \beta_\sigma$. We use the following point-isolating lemma to distinguish a from the remaining points in β_σ.

By Lemma 3.3, there exists a polynomial q of degree at most $d - 3$ such that

$$q(a) = 1 \qquad \text{and} \qquad q(b) = 0 \quad \text{for all } b \in \beta_\sigma \setminus \{a\}.$$

First we show that $a_0 = 1$. For every $i \in [n]$, the polynomial $q(x)b_i(x)$ is an $\mathbb{F}_2$-linear combination of equations in $\mathcal{B}$, so

$$0 = \sigma\big(q(x)b_i(x)\big) = \sum_{b \in \beta_\sigma} q(b)b_i(b) = a_i(a_i + a_0). \tag{1}$$

If $a_0 = 0$, then (1) forces $a_i = 0$ for every $i \in [n]$, contradicting the choice of the nonzero point a. Hence $a_0 = 1$.

With $a_0 = 1$, (1) says that the point a obeys the intended Booleanity constraints on the original coordinates, and the values $h_\ell(a)$ are the usual false-literal indicators of the Boolean assignment $(a_1, \ldots, a_n)$. Since φ is unsatisfiable, this assignment falsifies some clause, say C_j, and therefore $p_j(a) = 1$. Again $q(x)p_j(x)$ is an $\mathbb{F}_2$-linear combination of equations in $\mathcal{B}$, and hence

$$0 = \sigma\big(q(x)p_j(x)\big) = \sum_{b\in\beta_\sigma} q(b)p_j(b) = q(a)p_j(a) = 1,$$

a contradiction. Thus either $\beta_\sigma = \{\mathbf{0}\}$ or $|\beta_\sigma| \geq 2^{d-3}$. ◀

3.2 Obtaining Khot–Saket superposition soundness

We now define superposition satisfaction, a notion ruled out by the soundness analysis of Khot–Saket. We are particularly interested in this notion because assignments in superposition satisfaction naturally correspond to the factors in the symmetric rank-one decomposition given by Lemma 3.8.

▶ **Definition 3.5** (Superposition satisfaction and aggregate assignment)**.** *Let*

$$g(y) = \sum_{U,V} c_{U,V}\, y_U y_V + \sum_W d_W y_W$$

be a polynomial of degree at most 2 *and without constant term over* $\mathbb{F}_2$*. We say that assignments* $\tau^{(1)}, \ldots, \tau^{(t)}$ satisfy $g(y) = 0$ in superposition *if*

$$\sum_{i=1}^{t} g\big(\tau^{(i)}\big) = 0.$$

Their aggregate assignment *is the coordinate-wise sum*

$$\tau := \sum_{i=1}^{t} \tau^{(i)}.$$

For $\beta \subseteq \mathbb{F}_2^{n+1}$ and a polynomial f, write

$$\chi_\beta(f) := (-1)^{\sum_{a\in\beta} f(a)}.$$

The following correlation estimate is from [12, Lemma 3.3], which further builds on [6, Theorem 20].

▶ **Lemma 3.6** (Khot–Saket correlation bound; [12, Lemma 3.3])**.** *Let* $d \geq 8$ *be a positive multiple of* 4*. Let* σ *be a degree-*d *monomial assignment, and let* β_σ *be a minimum-cardinality set representing* σ *as in Lemma 3.2. Suppose* $|\beta_\sigma| \geq 2^{d-3}$*, and let* $\gamma, \alpha \subseteq \mathbb{F}_2^{n+1}$ *be arbitrary. Let* g *be a uniformly random multilinear polynomial of degree at most* $3d/4$*, and let* h *be a uniformly random multilinear polynomial of degree at most* $d/4$ *with zero constant term. Then*

$$\left|\mathop{\mathbb{E}}_{g,h}\big[\chi_{\beta_\sigma}(gh)\chi_\gamma(g)\chi_\alpha(h)\big]\right| \leq 2^{-2^{d/4-2}+1}.$$

▶ **Theorem 3.7** (Constant-free exact-superposition soundness, adapted from [12])**.** *There is a constant* $c > 0$ *such that the following holds. Let* $t \geq 1$*, and let* d *be a positive multiple of* 4 *with* $d \geq \max\{8, c\log(t+1)\}$*. From a* 3Sat *instance on* n *variables, one can construct in time* $n^{O(d)}$*, a system* $\mathcal{Q}$ *of quadratic equations with zero constant term, in variables*

$$\{y_S : S \in \mathcal{U}_{n,d}\}$$

with the following properties:

- *(YES) if the* 3Sat *instance is satisfiable, then $\mathcal{Q}$ has a satisfying assignment y with $y_{\{0\}} = 1$;*
- *(NO) if the* 3Sat *instance is unsatisfiable and $\tau^{(1)}, \dots, \tau^{(t)}$ satisfy all equations of $\mathcal{Q}$ in superposition, then their aggregate assignment $\tau := \sum_{i=1}^{t} \tau^{(i)}$ vanishes on every coordinate y_S, $S \in \mathcal{U}_{n,d}$.*

Moreover, $\mathcal{Q}$ is obtained by replacing each nonconstant monomial x^S in the system $\mathcal{B}$ from Theorem 3.4 with the variable y_S, and by adjoining the constraints

$$y_S y_T = y_{S \cup T} \qquad \text{for all } S, T \in \mathcal{U}_{n,d} \text{ with } |S \cup T| \le d.$$

Proof. Apply Theorem 3.4 to obtain the constant-free system $\mathcal{B}$ in the variables $x_0, x_1, \dots, x_n$. For every $S \in \mathcal{U}_{n,d}$ introduce a variable y_S, intended to represent the monomial x^S. Since every equation of $\mathcal{B}$ has zero constant term, no variable corresponding to $x^{\emptyset}$ is needed. The system $\mathcal{Q}$ contains:

- **(i)** for every equation $\sum_S c_S x^S = 0$ in $\mathcal{B}$, the linear equation $\sum_S c_S y_S = 0$ obtained by replacing each monomial x^S with y_S;
- **(ii)** every multiplicativity constraint $y_S y_T = y_{S \cup T}$ with $S, T \in \mathcal{U}_{n,d}$ and $|S \cup T| \le d$.

The YES case follows directly from Theorem 3.4. If a is the satisfying assignment there, with $a_0 = 1$, then setting $y_S := a^S$ satisfies all linearized equations from $\mathcal{B}$ and all multiplicativity constraints. In particular $y_{\{0\}} = 1$.

For the NO case, let $\tau^{(1)}, \dots, \tau^{(t)}$ be assignments to the variables y_S satisfying all equations of $\mathcal{Q}$ in superposition. Write

$$\tau_S^{(i)} := \tau^{(i)}(y_S), \qquad \tau_S := \sum_{i=1}^{t} \tau_S^{(i)} \qquad (S \in \mathcal{U}_{n,d}).$$

Thus $\tau = (\tau_S)_{S \in \mathcal{U}_{n,d}}$ is the aggregate y-assignment. Define the associated degree-d monomial assignments σ and $\sigma^{(i)}$ by

$$\sigma(x^S) := \tau_S, \qquad \sigma^{(i)}(x^S) := \tau_S^{(i)} \qquad (S \in \mathcal{U}_{n,d}),$$

extending them to polynomials by the constant-preserving convention from Lemma 3.2. Thus τ lives on the y-coordinates, while σ is the corresponding assignment to the x-monomials.

The main idea here is that, since the equations inherited from $\mathcal{B}$ are linear and homogeneous, superposition satisfaction forces the aggregate assignment $\tau = \sum_i \tau^{(i)}$ to satisfy these equations. Viewing the y-coordinates of τ as a monomial assignment σ, Theorem 3.4 then gives a dichotomy: either $\beta_\sigma = \{\mathbf{0}\}$, or β_σ is large. The latter possibility is ruled out by converting the superposition satisfaction of product equations $y_S y_T = y_{S \cup T}$ into a low-degree correlation identity, and applying the Khot–Saket correlation bound (Lemma 3.6).

The equations inherited from $\mathcal{B}$ become linear after the monomial-to-variable replacement in (i), and they have no constant term. Therefore superposition satisfaction implies that τ satisfies every such linearized equation. Equivalently, σ satisfies every equation of $\mathcal{B}$. By Theorem 3.4, either $\beta_\sigma = \{\mathbf{0}\}$ or $|\beta_\sigma| \ge 2^{d-3}$. If $\beta_\sigma = \{\mathbf{0}\}$, then for every $S \in \mathcal{U}_{n,d}$,

$$\tau_S = \sigma(x^S) = x^S(\mathbf{0}) = 0,$$

which is exactly the claimed vanishing of every coordinate y_S.

It remains to rule out the case $|\beta_\sigma| \ge 2^{d-3}$. For each i, let $\beta_i := \beta_{\sigma^{(i)}}$. The multiplicativity constraint $y_S y_T = y_{S \cup T}$, satisfied in superposition, gives

$$\sigma(x^{S \cup T}) = \tau_{S \cup T} = \sum_{i=1}^{t} \tau_S^{(i)} \tau_T^{(i)} = \sum_{i=1}^{t} \sigma^{(i)}(x^S) \sigma^{(i)}(x^T) \tag{2}$$

whenever $S, T \in \mathcal{U}_{n,d}$ and $|S \cup T| \leq d$. Using the same constant-preserving extension, (2) implies, by checking monomial pairs and extending $\mathbb{F}_2$-bilinearly, that

$$\sigma(gh) = \sum_{i=1}^{t} \sigma^{(i)}(g)\sigma^{(i)}(h) \tag{3}$$

for every polynomial g of degree at most $3d/4$ and every polynomial h of degree at most $d/4$ with zero constant term. Indeed, since h has zero constant term, it suffices to check $h = x^T$ and $g = 1$ or $g = x^S$. The case $g = x^S$ is exactly (2); the case $g = 1$ is the coordinate identity $\tau_T = \sum_i \tau_T^{(i)}$. The degree bounds ensure $|S \cup T| \leq d$ in the first case.

To analyze (3) using Fourier analysis, we need to switch from $\mathbb{F}_2$ to the real values in $\{-1, 1\}$. For bits $u, v \in \mathbb{F}_2$, if $a = (-1)^u$ and $b = (-1)^v$, then

$$(-1)^{uv} = a \wedge b := \frac{1 + a + b - ab}{2}.$$

Taking signs in (3) and using Lemma 3.2, we obtain, for every such pair g, h,

$$\chi_{\beta_\sigma}(gh) \prod_{i=1}^{t} (\chi_{\beta_i}(g) \wedge \chi_{\beta_i}(h)) = 1. \tag{4}$$

Now choose g and h uniformly at random from the two polynomial spaces above. The expectation of the left-hand side of (4) is 1. On the other hand, expanding each factor

$$a \wedge b = \frac{1 + a + b - ab}{2}$$

expresses the same expectation as a sum of at most 4^t terms, each with coefficient of absolute value at most 2^{-t} and each of the form

$$\mathop{\mathbb{E}}_{g,h}\left[\chi_{\beta_\sigma}(gh)\chi_\gamma(g)\chi_\alpha(h)\right]$$

for some subsets $\gamma, \alpha \subseteq \mathbb{F}_2^{n+1}$. By Lemma 3.6 above, the absolute value of the whole expectation is at most

$$2^t \cdot 2^{-2^{d/4-2}+1}.$$

Choosing the absolute constant c sufficiently large makes this quantity strictly smaller than 1 for every $t \geq 1$, contradicting (4). Hence the case $|\beta_\sigma| \geq 2^{d-3}$ cannot occur, and the aggregate assignment must vanish on every coordinate y_S. ◀

3.3 Linearizing the superposition instance

Fix a degree parameter d, and let $\mathcal{Q}$ be the quadratic system from Theorem 3.7. Its variables are $\{y_S : S \in \mathcal{U}_{n,d}\}$. We define a linear subspace of matrices with rows and columns indexed by $\mathcal{U}_{n,d}$ by imposing two kinds of linear constraints.

(1) *Equal-union constraints:*

$$A_{S,T} = A_{S',T'} \qquad \text{whenever } S \cup T = S' \cup T'. \tag{5}$$

(2) *Equation constraints:* for every equation of $\mathcal{Q}$ in the form

$$\sum_{S,T \in \mathcal{U}_{n,d}} c_{S,T} y_S y_T + \sum_{R \in \mathcal{U}_{n,d}} b_R y_R = 0, \tag{6}$$

where $c_{S,T}, b_R \in \mathbb{F}_2$, impose

$$\sum_{S,T \in \mathcal{U}_{n,d}} c_{S,T} A_{S,T} + \sum_{R \in \mathcal{U}_{n,d}} b_R A_{R,R} = 0. \tag{7}$$

Let $\mathcal{L}_d(\mathcal{Q})$ be the set of matrices satisfying (5) and (7). This is a linear subspace of $\mathbb{F}_2^{N\times N}$, where

$$N = |\mathcal{U}_{n,d}| = \sum_{j=1}^{d} \binom{n+1}{j}.$$

Furthermore, $\mathcal{L}_d(\mathcal{Q})$ only contains symmetric matrices due to the constraints $A_{S,T} = A_{T,S}$ in (5).

▶ **Lemma 3.8** (Decomposition into symmetric rank-one matrices, [12]). *Let $A \in \mathbb{F}_2^{N\times N}$ be symmetric of rank k. Then there is an integer t with $0 \le t \le \lfloor 3k/2 \rfloor$ and vectors $u^{(1)}, \dots, u^{(t)} \in \mathbb{F}_2^N$ such that*

$$A = \sum_{i=1}^{t} u^{(i)} u^{(i)\top}.$$

A proof of Lemma 3.8 can be found in the full version of this paper.

▶ **Lemma 3.9** (Low rank gives superposition assignments). *Let $k \ge 1$. Let $A \in \mathcal{L}_d(\mathcal{Q})$ have rank at most k, and let $t \ge \lfloor 3k/2 \rfloor$. Then there exist $u^{(1)}, \dots, u^{(t)} \in \mathbb{F}_2^N$ such that*

$$A = \sum_{i=1}^{t} u^{(i)} u^{(i)\top},$$

and the assignments $y_S \mapsto u_S^{(i)}$ satisfy every equation of $\mathcal{Q}$ in superposition.

Proof. Since $A \in \mathcal{L}_d(\mathcal{Q})$ satisfies the equal-union constraints, it is symmetric. By Lemma 3.8, A is a sum of at most $\lfloor 3k/2 \rfloor \le t$ symmetric rank-one matrices. Padding with zero vectors if necessary, write

$$A = \sum_{i=1}^{t} u^{(i)} u^{(i)\top}.$$

Fix an equation of $\mathcal{Q}$ written as in (6). Since A satisfies the corresponding equation constraint (7), we have

$$0 = \sum_{S,T\in\mathcal{U}_{n,d}} c_{S,T} A_{S,T} + \sum_{R\in\mathcal{U}_{n,d}} b_R A_{R,R}.$$

Substituting the rank-one decomposition and using $(u_R^{(i)})^2 = u_R^{(i)}$ over $\mathbb{F}_2$ gives

$$\begin{aligned} 0 &= \sum_{i=1}^{t} \left(\sum_{S,T\in\mathcal{U}_{n,d}} c_{S,T} u_S^{(i)} u_T^{(i)} + \sum_{R\in\mathcal{U}_{n,d}} b_R (u_R^{(i)})^2 \right) \\ &= \sum_{i=1}^{t} \left(\sum_{S,T\in\mathcal{U}_{n,d}} c_{S,T} u_S^{(i)} u_T^{(i)} + \sum_{R\in\mathcal{U}_{n,d}} b_R u_R^{(i)} \right). \end{aligned}$$

The last expression is the superposition sum of this equation evaluated on the assignments $y_S \mapsto u_S^{(i)}$. Hence the assignments satisfy every equation of $\mathcal{Q}$ in superposition. ◀

▶ **Lemma 3.10** (Zero aggregate assignment forces vanishing). *Let $A \in \mathcal{L}_d(\mathcal{Q})$, $A \neq 0$, admit a decomposition*

$$A = \sum_{i=1}^{t} u^{(i)} u^{(i)\top}$$

such that its aggregate assignment is zero:

$$\sum_{i=1}^{t} u_U^{(i)} = 0 \qquad \text{for every } U \in \mathcal{U}_{n,d}.$$

Then

$$\operatorname{rank}(A) \geq \binom{d+1}{\lfloor (d+1)/2 \rfloor}.$$

Proof. The zero-aggregation condition gives

$$A_{U,U} = \sum_{i=1}^{t} (u_U^{(i)})^2 = \sum_{i=1}^{t} u_U^{(i)} = 0, \qquad \forall U \in \mathcal{U}_{n,d}.$$

If $U, V \in \mathcal{U}_{n,d}$ and $|U \cup V| \leq d$, then $U \cup V \in \mathcal{U}_{n,d}$, and the equal-union constraints give

$$A_{U,V} = A_{U \cup V, U \cup V} = 0. \tag{8}$$

Suppose $A \neq 0$, and choose U, V with $A_{U,V} = 1$ minimizing $s := |U \cup V|$. By (8), we have $d < s \leq 2d$. Put $R := U \cup V$. For every pair $U', V' \in \mathcal{U}_{n,d}$ with $U' \cup V' = R$, the equal-union constraints give $A_{U',V'} = A_{U,V} = 1$.

Let

$$\mathcal{F} := \{F \subseteq R : |F| = \lfloor s/2 \rfloor\}, \qquad \mathcal{G} := \{G \subseteq R : |G| = \lceil s/2 \rceil\}.$$

Because $d < s \leq 2d$, every set in $\mathcal{F} \cup \mathcal{G}$ lies in $\mathcal{U}_{n,d}$. In the submatrix $A|_{\mathcal{F},\mathcal{G}}$, the entry indexed by (F, G) is 1 exactly when $G = R \setminus F$: in that case $F \cup G = R$, while all other pairs have union a proper subset of R and hence have entry 0 by the minimality of s. Thus $A|_{\mathcal{F},\mathcal{G}}$ is a permutation matrix, so

$$\operatorname{rank}(A) \geq \binom{s}{\lfloor s/2 \rfloor} \geq \binom{d+1}{\lfloor (d+1)/2 \rfloor}. \qquad \blacktriangleleft$$

▶ **Theorem 3.11** (Rank gap over $\mathbb{F}_2$). *For every integer $k \geq 1$, there is a deterministic $n^{O(\log k)}$-time reduction from a* 3SAT *instance on n variables to a linear subspace*

$$\mathcal{L} \subseteq \mathbb{F}_2^{N \times N}, \qquad N = n^{O(\log k)},$$

such that satisfiable instances yield a nonzero rank-1 matrix in $\mathcal{L}$, and unsatisfiable instances yield no nonzero matrix in $\mathcal{L}$ of rank at most k.

Proof. Set

$$t := \left\lfloor \frac{3k}{2} \right\rfloor,$$

and choose d to be a positive multiple of 4 such that

$$d \geq \max\{8, c \log(t+1)\} \qquad \text{and} \qquad \binom{d+1}{\lfloor (d+1)/2 \rfloor} > k,$$

where c is the constant from Theorem 3.7. Then $d = O(\log k)$.

Apply Theorem 3.7 with this d and output $\mathcal{L} := \mathcal{L}_d(\mathcal{Q})$. The construction time is $n^{O(d)} = n^{O(\log k)}$, and

$$N = |\mathcal{U}_{n,d}| = \sum_{j=1}^{d} \binom{n+1}{j} = n^{O(d)} = n^{O(\log k)}.$$

In the YES case, take a satisfying Boolean assignment to the original formula, set $a_0 = 1$, and define $y_S := a^S$ for every $S \in \mathcal{U}_{n,2d}$. Then $H_d(y) = v_d(a)v_d(a)^\top$ is nonzero of rank 1. Since every equation of $\mathcal{Q}$ vanishes at $(a^S)_{S\in\mathcal{U}_{n,d}}$, the matrix $H_d(y)$ satisfies both the equal-union constraints and the equation constraints. Hence $H_d(y) \in \mathcal{L}$.

In the NO case, suppose $A \in \mathcal{L}$ is nonzero and $\mathrm{rank}(A) \le k$. By Lemma 3.9, there are exactly t assignments whose rank-one sum is A and which satisfy all equations of $\mathcal{Q}$ in superposition. By Theorem 3.7, their aggregate assignment is zero. Then Lemma 3.10 and the choice of d imply that $\mathrm{rank}(A) > k$. ◀

3.4 Extending the hardness from $\mathbb{F}_2$ to $\mathbb{F}_{2^r}$

The following descent lemma is the black-box field-extension step. It loses a factor of r in the rank parameter, which is why the base-field gap below is run with parameter rk. Due to the page limit, we defer its proof to the full version of this paper.

▶ **Lemma 3.12** (Rank descent). *Let $\mathbb{K} = \mathbb{F}_{2^r}$, and let $\mathcal{L} \subseteq \mathbb{F}_2^{N\times N}$ be a linear subspace. If $A \in \mathcal{L} \otimes_{\mathbb{F}_2} \mathbb{K}$ is nonzero and $\mathrm{rank}_{\mathbb{K}}(A) \le k$, then there exists a nonzero $B \in \mathcal{L}$ with*

$$\mathrm{rank}_{\mathbb{F}_2}(B) \le rk.$$

Proof of Theorem 3.1. Let $\mathbb{K} = \mathbb{F}_{2^r}$ and set $k_0 := rk$. Apply Theorem 3.11 with rank parameter k_0 to obtain $\mathcal{M} \subseteq \mathbb{F}_2^{N\times N}$, and output

$$\mathcal{L} := \mathcal{M} \otimes_{\mathbb{F}_2} \mathbb{K} \subseteq \mathbb{K}^{N\times N},$$

equivalently the same homogeneous linear equations interpreted over $\mathbb{K}$.

Completeness is preserved under field extension: a nonzero rank-1 matrix in $\mathcal{M}$ remains a nonzero rank-1 matrix in $\mathcal{L}$. For soundness, suppose that the input formula is unsatisfiable and that $A \in \mathcal{L}$ is nonzero with $\mathrm{rank}_{\mathbb{K}}(A) \le k$. By Lemma 3.12, there is a nonzero $B \in \mathcal{M}$ with $\mathrm{rank}_{\mathbb{F}_2}(B) \le rk = k_0$, contradicting the soundness of $\mathcal{M}$.

The running time and dimension are

$$n^{O(\log k_0)} = n^{O(\log(rk))}, \qquad N = n^{O(\log k_0)} = n^{O(\log(rk))}.$$

For fixed extension degree r, this is $n^{O(\log k)}$ time and $N = n^{O(\log k)}$. ◀

The inapproximability statement Theorem 1.1 is a canonical corollary of Theorem 3.1.

▶ Remark 3.13 (Toward arbitrary finite fields). It is natural to ask whether the $n^{O(\log k)}$ rank-gap reduction of this section can be obtained over an arbitrary fixed finite field $\mathbb{F}_q$. The main technical step to achieve this is replacing the binary low-degree long code used in the Dinur–Guruswami and Khot–Saket frameworks [6, 12] with a suitable q-ary analogue. In particular, carrying out the superposition-soundness argument over $\mathbb{F}_q$ requires extending the relevant low-error Reed–Muller testing and, crucially, generalizing the correlation bound of Lemma 3.6 to arbitrary finite fields.

4 An $n^{O(k)}$ Construction over any $\mathbb{F}_q$

In this section, we start with the canonical NP-hardness of Boolean QuadEq, and prove Theorem 4.2 by a direct moment-matrix construction.

▶ **Lemma 4.1** (Boolean QuadEq hardness, [8, 10]). *Let $\mathbb{F}_q$ be any finite field. Given squarefree quadratic polynomials $f_1, \dots, f_m$ over $\mathbb{F}_q$ in variables $x_1, \dots, x_n$, it is NP-hard to decide whether there exists $a \in \{0,1\}^n \subseteq \mathbb{F}_q^n$ such that $f_1(a) = \cdots = f_m(a) = 0$. Equivalently, each input polynomial may be written as*

$$f_\ell(x) = \sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U} x^U, \qquad c_{\ell,U} \in \mathbb{F}_q,$$

where the coordinate indexed by $\emptyset$ is the constant term.

▶ **Theorem 4.2.** *Fix a finite field $\mathbb{F}_q$. For every integer $k \geq 1$, there is a deterministic $n^{O(k)}$-time reduction from a Boolean QuadEq instance with n variables over $\mathbb{F}_q$ to a linear subspace*

$$\mathcal{L} \subseteq \mathbb{F}_q^{N \times N}, \qquad N = n^{O(k)},$$

such that:

- *(YES) if the Boolean QuadEq instance is satisfiable, then $\mathcal{L}$ contains a nonzero matrix of rank 1;*
- *(NO) if the Boolean QuadEq instance is unsatisfiable, then $\mathcal{L}$ contains no nonzero matrix of rank at most k.*

Throughout this section all vector spaces, matrices, and polynomials are over a fixed finite field $\mathbb{F}_q$. We use the Boolean monomial and pseudo-moment conventions from Section 2; in particular, $\mathcal{V}_{n,d}$ includes the empty set.

4.1 The pseudo-moment subspace

Fix an integer $d \geq 1$. Following the notation of Section 2, a degree-$2d$ pseudo-moment vector is $y = (y_S)_{S \in \mathcal{V}_{n,2d}}$, and its associated pseudo-moment matrix is $H_d(y)$.

We define a linear subspace of matrices with rows and columns indexed by $\mathcal{V}_{n,d}$ by imposing two kinds of linear constraints.

(1) *Equal-union constraints:*

$$A_{S,T} = A_{S',T'} \qquad \text{whenever } S \cup T = S' \cup T'. \tag{9}$$

After imposing these constraints, for every $R \in \mathcal{V}_{n,2d}$ we write y_R for the common value $A_{S,T}$ over all pairs $S, T \in \mathcal{V}_{n,d}$ with $S \cup T = R$. For convenience, we describe the next set of constraints in terms of y.

(2) *Localizing constraints:* For each QuadEq equation

$$f_\ell(x) = \sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U} x^U, \qquad c_{\ell,U} \in \mathbb{F}_q,$$

we impose the linear constraints

$$\sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U}\, y_{U \cup W} = 0 \qquad \text{for every } \ell \in [m] \text{ and every } W \in \mathcal{V}_{n,2d-2}. \tag{10}$$

These are the moment analogues of the equations $x^W f_\ell(x) = 0$: after multiplying f_ℓ by any squarefree monomial of degree at most $2d-2$, the corresponding linear combination of pseudo-moments must vanish.

Let $\mathcal{L}_d(f_1, \ldots, f_m)$ denote the set of matrices satisfying the constraints above. This is a linear subspace of $\mathbb{F}_q^{N \times N}$, where

$$N = |\mathcal{V}_{n,d}| = \sum_{j=0}^{d} \binom{n}{j} = n^{O(d)}.$$

For completeness, if $a = (a_1, \ldots, a_n) \in \{0,1\}^n$ satisfies all equations $f_\ell(a) = 0$, set $y_S := a^S$ for every $S \in \mathcal{V}_{n,2d}$. Then $H_d(y) = v_d(a)v_d(a)^\top$ has rank 1 and is nonzero because the coordinate indexed by $\emptyset$ is 1. Moreover, for every $W \in \mathcal{V}_{n,2d-2}$,

$$\sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U} y_{U \cup W} = a^W f_\ell(a) = 0,$$

so $H_d(y) \in \mathcal{L}_d(f_1, \ldots, f_m)$.

We now begin the soundness proof. It suffices to show that every nonzero matrix in $\mathcal{L}_d(f_1, \ldots, f_m)$ with rank at most d yields an actual Boolean solution of the source instance. Let y be the unique pseudo-moment vector associated with such a matrix A, so $A = H_d(y)$ and y satisfies (10). For a genuine Boolean point a, multiplying the label of a monomial column by x_i multiplies that column by the scalar a_i; equivalently, the column space is invariant under multiplication by each variable. We will find two consecutive levels where low rank forces a similar form of invariance.

For $0 \le e \le d$, let

$$H_e(y) := \big(y_{S \cup T}\big)_{S,T \in \mathcal{V}_{n,e}}, \qquad r_e := \operatorname{rank} H_e(y),$$

and let C_e denote the column space of $H_e(y)$.

The soundness argument has four steps.

(1) Low rank forces a nonzero *flat level*, meaning a level e with $r_e = r_{e+1} > 0$.

(2) At such a flat level, multiplying a basis column by any variable x_i creates no new column direction. This defines a linear map $T_i : C_e \to C_e$, interpreted as multiplication by x_i.

(3) Because the relevant column relations are certified one level up (a.k.a inside C_{e+1}), these maps satisfy the Boolean multiplication rules $T_i^2 = T_i$ and $T_iT_j = T_jT_i$, and the localizing equations become the operator identities $f_\ell(T) = 0$.

(4) The commuting projections T_i have a common eigenvector. Their eigenvalues form a Boolean point $a \in \{0,1\}^n \subseteq \mathbb{F}_q^n$, and the identities $f_\ell(T) = 0$ force $f_\ell(a) = 0$.

4.2 Finding a nonzero flat level

Note that the ranks $r_0, r_1, \ldots, r_d$ are non-decreasing, and the final rank is at most d. Monotonicity alone would still allow the ranks to be zero for many initial levels and then grow by one at each remaining level. The key point is that a nonzero pseudo-moment cannot start growing this slowly. If S is a minimum-size set with $y_S \neq 0$, then the same permutation-submatrix idea as in Lemma 3.10 already forces, at level $s = \lceil |S|/2 \rceil$, the lower bound $r_s \ge s + 1$ (indeed $r_s \ge 2^{\Omega(s)}$ except for the smallest cases). If every subsequent positive step were strict, then

$$r_d \ge r_s + (d - s) \ge (s + 1) + (d - s) = d + 1,$$

contradicting $r_d \le d$. Thus some nonzero plateau $r_e = r_{e+1}$ must occur.

▶ **Lemma 4.3.** *Let $d \geq 1$. Suppose $H_d(y)$ is nonzero and* $\operatorname{rank} H_d(y) \leq d$. *Then there exists $e \in \{0, 1, \ldots, d-1\}$ such that*

$$r_e = r_{e+1} > 0.$$

Proof. Choose a set $S \in \mathcal{V}_{n,2d}$ of minimum size such that $y_S \neq 0$, and put

$$\lambda := y_S, \qquad h := |S|, \qquad s := \lceil h/2 \rceil .$$

Since $h \leq 2d$, we have $s \leq d$. The first step is the following rank lower bound forced by the first nonzero moment:

$$r_s \geq s + 1. \tag{11}$$

If $h = 0$, then $s = 0$ and $y_\emptyset = \lambda \neq 0$, so $r_s = r_0 = 1 = s + 1$. If $h = 1$, say $S = \{i\}$, then the minimality of S gives $y_\emptyset = 0$ and $y_S = \lambda$. The submatrix of $H_1(y)$ with rows and columns indexed by $\emptyset, S$ is

$$\begin{pmatrix} y_\emptyset & y_S \\ y_S & y_S \end{pmatrix} = \begin{pmatrix} 0 & \lambda \\ \lambda & \lambda \end{pmatrix},$$

which has rank $2 = s + 1$.

It remains to consider $h \geq 2$. Use the subsets $A \subseteq S$ of size $\lfloor h/2 \rfloor$ as row labels and the subsets $B \subseteq S$ of size $\lceil h/2 \rceil = s$ as column labels. The resulting submatrix of $H_s(y)$ has entries $y_{A \cup B}$. By the minimality of S, this entry is λ exactly when $A \cup B = S$, and is 0 otherwise. For each row A, there is a unique such column, namely $B = S \setminus A$. Thus this submatrix is λ times a permutation matrix of size

$$\binom{h}{\lfloor h/2 \rfloor}.$$

Therefore

$$r_s \geq \binom{h}{\lfloor h/2 \rfloor} \geq h \geq \lceil h/2 \rceil + 1 = s + 1,$$

where the middle inequality uses $h \geq 2$. This proves (11) in all cases.

The matrices $H_0(y), H_1(y), \ldots, H_d(y)$ are nested principal submatrices, so the ranks $r_0, r_1, \ldots, r_d$ are nondecreasing. Suppose, for contradiction, that no $e \in \{0, 1, \ldots, d-1\}$ satisfies $r_e = r_{e+1} > 0$. Since $r_s > 0$ and the ranks are nondecreasing, the ranks strictly increase at every such step. Thus

$$r_d \geq r_s + (d - s) \geq (s + 1) + (d - s) = d + 1,$$

contradicting $r_d = \operatorname{rank} H_d(y) \leq d$. Therefore a nonzero flat level exists. ◀

4.3 Multiplication operators and localizing identities

For $0 \leq e \leq d$ and a set $A \subseteq [n]$ with $|A| \leq 2d - e$, define the truncated column

$$\mathbf{c}_e(A) := \big(y_{R \cup A}\big)_{R \in \mathcal{V}_{n,e}} \in \mathbb{F}_q^{\mathcal{V}_{n,e}}.$$

If $A \in \mathcal{V}_{n,e}$, then $\mathbf{c}_e(A)$ is exactly the column of $H_e(y)$ indexed by A. If $|A| = t > e$, it is not literally a column of $H_e(y)$, but it is still the degree-e truncation of the column indexed by A, in a larger moment matrix $H_t(y)$.

We now explain how the flat level is used. At a flat level $r_e = r_{e+1}$, any basis of the column space C_e remains a basis after lifting the columns to level $e+1$. Hence multiplying the column label by x_i gives a well-defined operator $T_i : C_e \to C_e$. Furthermore, this extra-level lift certifies that

$$T_i \mathbf{c}_e(A) = \mathbf{c}_e(A \cup \{i\})$$

for every $A \in \mathcal{V}_{n,e+1}$. This allows us to multiply the column label by a second variable consistently, yielding $T_i^2 = T_i$, $T_iT_j = T_jT_i$, and ultimately translating the localizing constraints into the operator identities $f_\ell(T) = 0$.

▶ **Lemma 4.4.** *Assume $r_e = r_{e+1}$ and $e \le d-1$. Let $\mathcal{I} \subseteq \mathcal{V}_{n,e}$ be such that the vectors $\{\mathbf{c}_e(B) : B \in \mathcal{I}\}$ form a basis for C_e. Then, for every $i \in [n]$, there is a linear map $T_i : C_e \to C_e$ satisfying*

$$T_i \mathbf{c}_e(B) = \mathbf{c}_e(B \cup \{i\}) \qquad \text{for every } B \in \mathcal{I}.$$

Moreover, for every $A \in \mathcal{V}_{n,e+1}$ and every $i \in [n]$, the vector $\mathbf{c}_e(A)$ lies in C_e and

$$T_i \mathbf{c}_e(A) = \mathbf{c}_e(A \cup \{i\}). \tag{12}$$

Proof. First observe that the longer vectors $\{\mathbf{c}_{e+1}(B) : B \in \mathcal{I}\}$ are linearly independent: any linear relation among them restricts, on the rows indexed by $\mathcal{V}_{n,e}$, to the same linear relation among the basis vectors $\{\mathbf{c}_e(B) : B \in \mathcal{I}\}$. Since $r_e = r_{e+1}$, these longer vectors form a basis for the column space of $H_{e+1}(y)$.

For every $i \in [n]$ and every $B \in \mathcal{I}$, the vector $\mathbf{c}_{e+1}(B \cup \{i\})$ is a column of $H_{e+1}(y)$, and hence has a unique expansion in the basis $\{\mathbf{c}_{e+1}(P) : P \in \mathcal{I}\}$. Restricting that expansion to rows of degree at most e shows that $\mathbf{c}_e(B \cup \{i\}) \in C_e$. We may therefore define T_i on the basis by

$$T_i \mathbf{c}_e(B) := \mathbf{c}_e(B \cup \{i\}) \qquad (B \in \mathcal{I}),$$

and extend linearly to all of C_e.

Now fix $A \in \mathcal{V}_{n,e+1}$. We would like to argue that $T_i \mathbf{c}_e(A) = \mathbf{c}_e(A \cup \{i\})$. Since $\mathbf{c}_{e+1}(A)$ is a column of $H_{e+1}(y)$, there are unique coefficients $\lambda_{A,B} \in \mathbb{F}_q$ such that

$$\mathbf{c}_{e+1}(A) = \sum_{B \in \mathcal{I}} \lambda_{A,B} \mathbf{c}_{e+1}(B).$$

Restricting to rows in $\mathcal{V}_{n,e}$ gives

$$\mathbf{c}_e(A) = \sum_{B \in \mathcal{I}} \lambda_{A,B} \mathbf{c}_e(B),$$

so $\mathbf{c}_e(A) \in C_e$. By linearity and the definition of T_i on the basis,

$$T_i \mathbf{c}_e(A) = \sum_{B \in \mathcal{I}} \lambda_{A,B} T_i \mathbf{c}_e(B) = \sum_{B \in \mathcal{I}} \lambda_{A,B} \mathbf{c}_e(B \cup \{i\}).$$

It remains to identify the last expression with $\mathbf{c}_e(A \cup \{i\})$. Fix a row label $R \in \mathcal{V}_{n,e}$. Since $R \cup \{i\} \in \mathcal{V}_{n,e+1}$, evaluating the expansion of $\mathbf{c}_{e+1}(\cdot)$ at the row $R \cup \{i\}$ gives

$$\begin{aligned}
\sum_{B \in \mathcal{I}} \lambda_{A,B} \big(\mathbf{c}_e(B \cup \{i\})\big)_R &= \sum_{B \in \mathcal{I}} \lambda_{A,B} y_{R \cup B \cup \{i\}} \\
&= \sum_{B \in \mathcal{I}} \lambda_{A,B} \big(\mathbf{c}_{e+1}(B)\big)_{R \cup \{i\}} \\
&= \big(\mathbf{c}_{e+1}(A)\big)_{R \cup \{i\}} \\
&= y_{R \cup A \cup \{i\}} = \big(\mathbf{c}_e(A \cup \{i\})\big)_R.
\end{aligned}$$

This holds for every row $R \in \mathcal{V}_{n,e}$, so $T_i \mathbf{c}_e(A) = \mathbf{c}_e(A \cup \{i\})$. The vector on the right is well-defined because $|A \cup \{i\}| \le e + 2 \le 2d - e$, using $e \le d - 1$. ◀

▶ **Lemma 4.5.** *Assume $r_e = r_{e+1} > 0$ and $e \le d-1$. Let C_e be the column space of $H_e(y)$. If y satisfies the equations* (10)*, then there are linear maps*

$$T_1, \dots, T_n : C_e \to C_e$$

with the following properties:

(i) *$T_i^2 = T_i$ for every i;*

(ii) *$T_i T_j = T_j T_i$ for every i, j;*

(iii) *for every $\ell \in [m]$,*

$$\sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U} T_U = 0 \qquad \text{as a linear map on } C_e,$$

where $T_U := \prod_{i \in U} T_i$ and $T_\emptyset$ is the identity map.

Proof. Choose $\mathcal{I} \subseteq \mathcal{V}_{n,e}$ such that $\{\mathbf{c}_e(B) : B \in \mathcal{I}\}$ is a basis for C_e, and define the maps T_i as in Lemma 4.4. By Lemma 4.4, for every $A \in \mathcal{V}_{n,e+1}$ and every i,

$$T_i \mathbf{c}_e(A) = \mathbf{c}_e(A \cup \{i\}). \tag{13}$$

We first prove the Boolean multiplication rules. Fix $B \in \mathcal{I}$ and $i, j \in [n]$. Taking $A = B$ in (13) gives $T_i \mathbf{c}_e(B) = \mathbf{c}_e(B \cup \{i\})$. Since $B \cup \{i\} \in \mathcal{V}_{n,e+1}$, applying (13) with $A = B \cup \{i\}$ and variable j gives

$$T_j T_i \mathbf{c}_e(B) = T_j \mathbf{c}_e(B \cup \{i\}) = \mathbf{c}_e(B \cup \{i, j\}). \tag{14}$$

Taking $j = i$ in (14) gives $T_i^2 \mathbf{c}_e(B) = T_i \mathbf{c}_e(B)$, because $B \cup \{i, i\} = B \cup \{i\}$. Since this holds on the basis vectors $\{\mathbf{c}_e(B) : B \in \mathcal{I}\}$, we get $T_i^2 = T_i$. Swapping i and j in (14) gives

$$T_i T_j \mathbf{c}_e(B) = \mathbf{c}_e(B \cup \{j, i\}) = \mathbf{c}_e(B \cup \{i, j\}) = T_j T_i \mathbf{c}_e(B),$$

again on every basis vector. Hence $T_i T_j = T_j T_i$.

It remains to translate the original equations into operator identities. Because the T_i commute, $T_U := \prod_{i \in U} T_i$ is well-defined. From (13) when $|U| \le 1$, and from (14) when $|U| = 2$, we have

$$T_U \mathbf{c}_e(B) = \mathbf{c}_e(B \cup U) \qquad \text{for every } B \in \mathcal{I} \text{ and every } U \in \mathcal{V}_{n,2}.$$

Therefore, for every row label $R \in \mathcal{V}_{n,e}$,

$$\left(\sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U} T_U \mathbf{c}_e(B) \right)_R = \sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U} y_{R \cup B \cup U}.$$

Here $R \cup B$ has size at most $2e$. Since $e \le d-1$, we have $2e \le 2d-2$, so the localizing equations (10) apply with $W = R \cup B$. Hence the last sum is 0. Thus $\sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U} T_U$ kills every basis vector of C_e, and therefore it is the zero map on C_e. ◀

4.4 Decoding a satisfying assignment from commuting projections

The next lemma records the common-eigenvector property for commuting projections. We omit the proof here.

▶ **Lemma 4.6.** *Let C be a nonzero finite-dimensional vector space over $\mathbb{F}_q$, and let $T_1, \dots, T_n$ be commuting linear maps on C with $T_i^2 = T_i$ for every i. Then there are $v \in C \setminus \{0\}$ and scalars $a_i \in \{0,1\} \subseteq \mathbb{F}_q$, $i \in [n]$, such that*

$$T_i v = a_i v \qquad \text{for every } i = 1, \dots, n.$$

▶ **Lemma 4.7.** *Let C be a nonzero finite-dimensional vector space over $\mathbb{F}_q$, and let $T_1, \dots, T_n$ be commuting linear maps on C. Suppose that, for every $\ell \in [m]$,*

$$\sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U} T_U = 0 \qquad \text{as a linear map on } C,$$

where $T_U := \prod_{i \in U} T_i$ and $T_\emptyset$ is the identity map. If there are $v \in C \setminus \{0\}$ and scalars $a_1, \dots, a_n \in \{0,1\} \subseteq \mathbb{F}_q$ such that $T_i v = a_i v$ for every i, then $a = (a_1, \dots, a_n)$ satisfies $f_1(a) = \cdots = f_m(a) = 0$.

Proof. Since the operators commute, for every $U \in \mathcal{V}_{n,2}$ we have

$$T_U v = a^U v.$$

Therefore, for every ℓ,

$$0 = \Big(\sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U} T_U \Big) v = \Big(\sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U} a^U \Big) v = f_\ell(a) v.$$

Since $v \neq 0$, the scalar $f_\ell(a)$ must be zero. Hence the Boolean point $a = (a_1, \dots, a_n)$ satisfies all equations. ◀

We are now ready to prove Theorem 4.2.

Proof of Theorem 4.2. Given the Boolean QuadEq instance $f_1, \dots, f_m$, set $d := k$ and output

$$\mathcal{L} := \mathcal{L}_d(f_1, \dots, f_m).$$

The matrix size is

$$N = |\mathcal{V}_{n,d}| = \sum_{j=0}^{k} \binom{n}{j} = n^{O(k)},$$

and the number of equal-union and localizing constraints is also $(n+m)n^{O(k)}$, so the construction has the claimed running time.

If the Boolean QuadEq instance has a solution, the completeness argument above shows that $\mathcal{L}$ contains a nonzero rank-1 matrix. If the Boolean QuadEq instance has no solution and $A \in \mathcal{L}$ is nonzero with $\operatorname{rank}(A) \le k$, then the unique pseudo-moment vector y associated with A satisfies $A = H_k(y)$ and (10). By Lemma 4.3, choose a flat level e with $r_e = r_{e+1} > 0$. By Lemma 4.5, the column space C_e carries commuting projections $T_1, \dots, T_n$, and for every $\ell \in [m]$,

$$\sum_{U \in \mathcal{V}_{n,2}} c_{\ell,U} T_U = 0.$$

By Lemma 4.6, there are a nonzero vector $v \in C_e$ and scalars $a_1, \dots, a_n \in \{0,1\} \subseteq \mathbb{F}_q$ such that $T_i v = a_i v$ for every i. By Lemma 4.7, a is a Boolean solution, a contradiction. Hence no such nonzero matrix exists. ◀

The inapproximability statement Theorem 1.2 is a standard corollary of Theorem 4.2.

References

1 Per Austrin and Subhash Khot. A simple deterministic reduction for the gap minimum distance of code problem. *IEEE Transactions on Information Theory*, 60(10):6636–6645, 2014. doi:10.1109/TIT.2014.2340869.

2 Boaz Barak and David Steurer. Sum-of-squares proofs and the quest toward optimal algorithms. In *Proceedings of the International Congress of Mathematicians (ICM)*, 2014. arXiv:1404.5236.

3 Vijay Bhattiprolu, Venkatesan Guruswami, Euiwoong Lee, and Xuandi Ren. Inapproximability of finding sparse vectors in codes, subspaces, and lattices. In *66th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2025, Sydney, Australia, December 14-17, 2025*, pages 1295–1303. IEEE, 2025. doi:10.1109/FOCS63196.2025.00068.

4 Qi Cheng and Daqing Wan. A deterministic reduction for the gap minimum distance problem. *IEEE Transactions on Information Theory*, 58(11):6935–6941, 2012. doi:10.1109/TIT.2012.2209198.

5 Nicolas T. Courtois. Efficient zero-knowledge authentication based on a linear algebra problem MinRank. In *Advances in Cryptology – ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 402–421. Springer, 2001. doi:10.1007/3-540-45682-1_24.

6 Irit Dinur and Venkatesan Guruswami. Pcps via the low-degree long code and hardness for constrained hypergraph coloring. *Israel Journal of Mathematics*, 209(2):611–649, 2015. doi:10.1007/s11856-015-1231-3.

7 Ilya Dumer, Daniele Micciancio, and Madhu Sudan. Hardness of approximating the minimum distance of a linear code. *IEEE Transactions on Information Theory*, 49(1):22–37, 2003. doi:10.1109/TIT.2002.806118.

8 Aviezri S. Fraenkel and Yaacov Yesha. Complexity of problems in games, graphs and algebraic equations. *Discrete Applied Mathematics*, 1(1–2):15–30, 1979. doi:10.1016/0166-218X(79)90012-X.

9 Philippe Gaborit and Gilles Zémor. On the hardness of the decoding and the minimum distance problems for rank codes. *IEEE Transactions on Information Theory*, 62(12):7245–7252, 2016. doi:10.1109/TIT.2016.2616127.

10 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

11 Sangxia Huang. $2^{(\log N)^{1/10-o(1)}}$ hardness for hypergraph coloring. *arXiv preprint arXiv:1504.03923*, 2015. doi:10.48550/arXiv.1504.03923.

12 Subhash Khot and Rishi Saket. Hardness of coloring 2-colorable 12-uniform hypergraphs with $2^{(\log n)^{\Omega(1)}}$ colors. In *55th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2014)*, pages 206–215, 2014. doi:10.1109/FOCS.2014.30.

13 Jean B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001. doi:10.1137/S1052623400366802.

14 Monique Laurent. Sums of squares, moment matrices and optimization over polynomials. In Mihai Putinar and Seth Sullivant, editors, *Emerging Applications of Algebraic Geometry*, volume 149 of *The IMA Volumes in Mathematics and its Applications*, pages 157–270. Springer, 2009. doi:10.1007/978-0-387-09686-5_7.

15 Daniele Micciancio. Locally dense codes. In *2014 IEEE 29th Conference on Computational Complexity (CCC)*, pages 90–97. IEEE, 2014. doi:10.1109/CCC.2014.17.

16 Pablo A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming*, 96:293–320, 2003. doi:10.1007/s10107-003-0387-5.

17 Alberto Ravagnani. Rank-metric codes and their duality theory. *Designs, Codes and Cryptography*, 80(1):197–216, 2016. doi:10.1007/s10623-015-0077-3.

On the Approximability of Parameterized Minimum Monotone Satisfying Assignment

Venkatesan Guruswami
Simons Institute for the Theory of Computing, University of California, Berkeley, CA, USA
Departments of EECS and Mathematics, University of California, Berkeley, CA, USA

Bingkai Lin
State Key Laboratory of Novel Software Technology, Nanjing University, China

Xuandi Ren
Department of EECS, University of California, Berkeley, CA, USA

Xin Zheng
State Key Laboratory of Novel Software Technology, Nanjing University, China

Abstract

The parameterized Minimum Monotone Satisfying Assignment (k-MMSA) problem asks whether a monotone Boolean circuit admits a satisfying assignment of Hamming weight at most k. The MMSA hierarchy is defined by allowing a bounded number of alternations between AND and OR gates in the circuit. While the polynomial-time approximability of the MMSA hierarchy has been studied extensively, much less is known in the parameterized setting. In particular, k-MMSA$_2$ is the well-known k-SetCover problem, whose parameterized inapproximability lies in the $\mathsf{polylog}(n)$ regime. In contrast, k-MMSA$_4$ captures k-MinLabel, for which known lower bounds give $\mathsf{poly}(n)$ inapproximability. Sandwiched by k-MMSA$_2$ and k-MMSA$_4$, the inapproximability of k-MMSA$_3$ remained comparatively unexplored.

In this paper, we give an FPT-time $O(2^k \log n)$-approximation algorithm for k-MMSA$_3$, suggesting that in the fixed-parameter regime, the third level of MMSA remains surprisingly close to the second level. Complementing this algorithm, we also give an FPT-time gap-preserving reduction from k-MMSA$_3$ to k-MMSA$_2$. Thus, stronger inapproximability for k-MMSA$_3$ would imply new hardness for k-MMSA$_2$, potentially offering a route around the current barriers for the latter problem.

Revisiting Marx's reduction from k-MMSA$_t$ to gap k-MMSA$_{t+2}$, we also show that k-MMSA$_4$ admits no $n^{o(1)}$-factor FPT approximation unless W[2]=FPT, and no $n^{O(1/k)}$-factor approximation running in $n^{o(k)}$ time under ETH. These results separate the parameterized approximability behavior of the third and fourth levels and clarify where stronger inapproximability enters the k-MMSA hierarchy.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms; Theory of computation → Approximation algorithms analysis; Theory of computation → Problems, reductions and completeness

Keywords and phrases Parameterized approximation, Minimum Monotone Satisfying Assignment, Set Cover, inapproximability

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.20

Category APPROX

Funding *Venkatesan Guruswami*: Research supported in part by NSF grant CCF-2211972, a DARPA grant under Contract No. HR0011262E031, and a Simons Investigator award.
Xuandi Ren: Supported in part by NSF grant CCF-2228287 and V.G.'s Simons Investigator award.

Acknowledgements We thank Yican Sun for helpful early discussions with the authors on the k-MMSA problem.


© Venkatesan Guruswami, Bingkai Lin, Xuandi Ren, and Xin Zheng;
licensed under Creative Commons License CC-BY 4.0



1 Introduction

Given a Boolean circuit C consisting of only $\wedge$ and $\vee$ gates on n input bits, the Minimum Monotone Satisfying Assignment (MMSA) problem asks to find a satisfying assignment of C with minimum Hamming weight.

MMSA was first introduced by Alekhnovich, Buss, Moran and Pitassi [1]. In a later work, Dinur and Safra [9] defined a hierarchy of MMSA problems by restricting the number of alternations of the monotone circuit[1]. Specifically, MMSA_t refers to the problem where the alternation starts with an $\wedge$, and has at most t levels. This hierarchy turns out to be equivalent to an AND/OR scheduling hierarchy suggested by Goldwasser and Motwani [15].

Looking into this hierarchy, MMSA_1 can be solved easily by simply outputting the width of the $\wedge$ gate. MMSA_2 is equivalent to the classical Set Cover problem, by identifying each set with a variable and each element to cover with an inner disjunction. Thus, MMSA_2 inherits the $O(\log n)$-approximation [16, 21, 7] and $(1-o(1))\ln n$ NP-inapproximability [8] of Set Cover. In [9], Dinur and Safra showed a $2^{\log^{1-o(1)} n}$ inapproximability for MMSA_3 via a reduction from PCP, and placed the Label-Cover problem[2] between MMSA_3 and MMSA_4. Recently, Chlamtáč, Makarychev, and Vakilian [6] revisited the polynomial-time approximability of this hierarchy. They gave an $\widetilde{O}(N^{1/3})$-approximation for MMSA_3 and MMSA_4, and more generally an $\widetilde{O}(N^{1-\delta_t})$-approximation for MMSA_t, where N is the total number of gates and variables, and $\delta_t = \frac{1}{3}\cdot 2^{3-\lceil t/2\rceil}$.

In a separate line of work, Umans [23] studied the *semantic* variant of Minimum Monotone Satisfying Assignment problem. In his formulation, the input is a circuit that is promised to accept a monotone set of strings. In contrast, [1, 9] studies the syntactic variant where the circuit itself is required to be monotone. Umans showed the NP-hardness of an $n^{1/5-\varepsilon}$-approximation for the semantic problem.

In the parameterized world, there is even more connection between MMSA and other important problems. Parameterized by the target Hamming weight k, the unbounded-alternation circuit version of k-MMSA is $\mathsf{W[P]}$-complete, whereas the corresponding formula version is $\mathsf{W[SAT]}$-complete; imposing constant-depth and weft-t restrictions on weighted circuit satisfiability yields the usual $\mathsf{W}[t]$-hierarchy [13, 11]. As for the bounded-depth variants, k-MMSA_2 is equivalent to k-Setcover, the canonical $\mathsf{W}[2]$-complete problem [10, 13, 11]. Inapproximability for k-Setcover was known under different complexity assumptions: there are no $n^{o(k)}$ time (respectively, $n^{k-\varepsilon}$ time) $(\frac{\log n}{\log\log n})^{1/k}$-approximation under ETH (respectively, SETH) [19]; no FPT time $(\log n)^{o(1)}$-approximation under $\mathsf{W}[1]\neq\mathsf{FPT}$ [17, 19][3]; and no FPT time constant approximation under $\mathsf{W}[2]\neq\mathsf{FPT}$ [20].

Despite these substantial results, the parameterized inaprproximability of k-Setcover is not yet fully understood. Several central questions remain open and have been raised repeatedly in the literature [12, 17, 19, 18, 20]. Can the FPT-time inapproximability factor be strengthened to match the $O(\log n)$-approximation by the greedy algorithm? Can the total-FPT-inapproximability[4] be based on the minimal assumption $\mathsf{W}[2]\neq\mathsf{FPT}$? Under

[1] Strictly speaking, [9] studied monotone *formulae* rather than general *circuits*. For fixed depth t, the two are interconvertible with polynomial blow-up. The distinction matters only in the unbounded-depth setting, e.g., in our later discussion on $\mathsf{W[P]}/\mathsf{W[SAT]}$-completeness of unbounded-alternation k-MMSA.

[2] This is the minimization Label-Cover problem in the Arora–Lund sense [3], rather than the Projection-game formulation, where the value is the maximum fraction of satisfied constraints.

[3] The factor in the statement of [17] is $(\log n)^{1/\mathsf{poly}(k)}$, but by a simple padding trick, we can actually get any FPT-time $(\log n)^{o(1)}$-factor inapproximability. See Lemma 2.8 for a formal discussion.

[4] By total-FPT-inapproximability, we mean ruling out FPT-time $f(k)$-approximation algorithms for every computable function f.

ETH, or even Gap-ETH, can one go beyond the current $(\log n)^{1/k}$-scale barrier for $n^{o(k)}$-time inapproximability? These questions motivate the study of k-MMSA$_3$, the closest natural extension of k-MMSA$_2$ in the k-MMSA hierarchy. In particular, we ask whether the additional expressive power of k-MMSA$_3$ enables reductions that bypass the known barriers for k-SetCover, and whether the techniques developed for k-MMSA$_3$ can in turn shed light on k-SetCover itself.

We remark that Dinur and Safra's approach [9] for non-parameterized MMSA$_3$ relies on building one Boolean variable $L[x, a]$ for each (variable, value) pair in the PCP construction. In the yes case, selecting exactly one value per PCP variable yields a satisfying assignment of weight n, where n is the number of variables. However, this approach does not extend to the parameterized setting: to keep the solution weight bounded by k, one would need to start from a *parameterized* CSP with only k variables, but for such CSPs, even just greedily satisfying the constraints adjacent to the maximum degree variable gives a $k^{O(1)}$ factor approximation[5], which precludes deriving any inapproximability factor that grows with n.

With an extra level of alternation, k-MMSA$_4$ possesses a stronger inapproximability. In his paper studying parameterized monotone and antimonotone circuit satisfiability, Marx [22] gave a gap-creating reduction from any k-MMSA$_t$ to k-MMSA$_{t+2}$, and thus ruled out any FPT-time $f(k)$-approximation for k-MMSA$_4$ under W[2] $\neq$ FPT. We remark that k-MMSA$_4$ also captures the k-MinLabel problem defined in [4]. It was known in [4] that, under Gap-ETH, any $f(k)$-approximation of k-MinLabel requires $n^{\Omega(k)}$ time. It was also implicit in [17, 18] that, k-MinLabel is hard to approximate within an $n^{o(1)}$-factor[6] in FPT time under W[1] $\neq$ FPT, or an $n^{1/\mathrm{poly}(k)}$-factor in $n^{o(k)}$ time under ETH, via reductions from k-MaxCover.

Our Contributions

In this paper, we give an FPT time $O(2^k \log n)$-approximation algorithm for k-MMSA$_3$ (Theorem 3.1), which shows, in contrast to the non-parameterized world, that approximating k-MMSA$_3$ is not significantly harder than approximating k-MMSA$_2$. On the hardness side, we give a reduction from (k, h)-gap k-MMSA$_3$ to $(2^k, h/k)$-gap k-MMSA$_2$ (Theorem 4.1). This connection has two implications. First, it helps explain why stronger inapproximability for k-MMSA$_3$ has not appeared separately from k-MMSA$_2$. Second, via this reduction, any sufficiently strong lower bound for k-MMSA$_3$ would yield new FPT inapproximability results for k-SetCover. In particular, a $(\log n)^{\Omega(1)}$-factor FPT inapproximability for k-MMSA$_3$ under W[1] $\neq$ FPT would imply the same for k-SetCover. Similarly, under W[2] $\neq$ FPT, an $\omega(2^k)$ FPT inapproximability for k-MMSA$_3$ would translate into a superconstant FPT inapproximability for k-SetCover, resolving long standing open problems.

We also revisit Marx's reduction [22] and show that it yields stronger consequences than previously recognized. By carefully picking the parameters, we show k-MMSA$_4$ is hard to approximate within an $n^{o(1)}$-factor[6] in FPT time under W[2] $\neq$ FPT, and hard to approximate within an $n^{O(1/k)}$-factor in $n^{o(k)}$ time under ETH, improving the previous $f(k)$-factor [22] and $n^{1/\mathrm{poly}(k)}$-factor [17, 18], respectively.

We summarize the (in)approximability results for parameterized MMSA hierarchy in Table 1.

[5] We may assume without loss of generality that the parameterized CSP has arity 2. Arity-q CSPs reduce to arity 2 by the standard label-coverization trick, with only an $O(q)$-factor loss in the gap.
[6] Requires padding, see Lemma 2.8.

Table 1 Approximability and Inapproximability Results for Parameterized k-MMSA.

Problems	Approximability			Inapproximability			
	Ratio	Runtime	Reference	Ratio	Runtime	Assumption	Reference
k-MMSA$_2$	$O(\log n)$	poly(n)	[16, 21] [7]	$\left(\frac{\log n}{\log\log n}\right)^{1/k}$	$n^{o(k)}$	ETH	[19]
				$(\log n)^{1/\text{poly}(k)}$			[17]
				$(\log n)^{o(1)}$	FPT	W[1] $\neq$ FPT	[17, 19]
				any $O(1)$		W[2] $\neq$ FPT	[20]
k-MMSA$_3$	$O(2^k \log n)$	FPT	Theorem 3.1	Inherited from above			
k-MMSA$_4$	n	poly(n)	Trivial	$n^{1/\text{poly}(k)}$	$n^{o(k)}$	ETH	[17, 18]
				$n^{O(1/k)}$			Corollary 5.6
				$n^{o(1)}$	FPT	W[1] $\neq$ FPT	[17, 18]
				any $f(k)$		W[2] $\neq$ FPT	[22]
				$n^{o(1)}$			Corollary 5.7

2 Preliminaries

2.1 Perfect Hash Families

▶ **Definition 2.1** ((n, m, Σ, h)-perfect hash family [14]). *An (n, m, Σ, h)-perfect hash family is a family of m functions $\mathcal{F} = \{f_i : [n] \to \Sigma \mid i \in [m]\}$, such that for every subset $T \subseteq [n]$ where $|T| \leq h$, there is a function $f \in \mathcal{F}$ such that*

$$\forall x, y \in T \text{ with } x \neq y,\ f(x) \neq f(y).$$

In other words, $\mathcal{F}$ is an (n, m, Σ, h)-perfect hash family if for any h distinct elements in $[n]$, there is a function $f \in \mathcal{F}$ which maps them to h distinct values.

When $|\Sigma| \geq h^2$, an (n, m, Σ, h)-perfect hash family can be constructed efficiently and deterministically:

▶ **Lemma 2.2** ([2]). *An (n, m, Σ, h)-perfect hash family with $|\Sigma| = h^2, m = h^{O(1)} \log n$ can be constructed deterministically in time $h^{O(1)} \cdot n \log n$.*

2.2 Parameterized Problems

We begin with a definition of the parameterized version of the MMSA problem.

▶ **Definition 2.3** (k-MMSA$_t$). *For any $t, k \in \mathbb{N}^+$, k-MMSA$_t$ asks whether there is a satisfying assignment with Hamming weight at most k, for a monotone circuit C of the following form.*

$$C(x_1, \ldots, x_n) = \begin{cases} \bigwedge_{i_1=1}^{n} \bigvee_{i_2=1}^{n} \cdots \bigwedge_{i_t=1}^{n} x_{w_{i_1,\ldots,i_t}}, & t \text{ is odd}, \\ \bigwedge_{i_1=1}^{n} \bigvee_{i_2=1}^{n} \cdots \bigwedge_{i_{t-1}=1}^{n} \bigvee_{i_t=1}^{n} x_{w_{i_1,\ldots,i_t}}, & t \text{ is even}, \end{cases}$$

where for every $i_1, \ldots, i_t \in [n]$, $w_{i_1,\ldots,i_t}$ is an index in $[n]$[7].

[7] We generally allow the width of each gate to be polynomial in the number of input bits, but for simplicity, we can pad unused bits and thus use n for both of them.

We write $\mathrm{SAT}(C)$ *to denote the set of satisfying assignments of* C*, and write* $\mathrm{OPT}(C)$ *to denote the minimum weight of any satisfying assignment.*

For any $h > k$*, we write* (k, h)*-gap MMSA*$_t$ *for the task of distinguishing between* $\mathrm{OPT}(C) \le k$ *and* $\mathrm{OPT}(C) > h$*, or equivalently, finding a satisfying assignment of* C *with weight at most* h *given the promise that* $\mathrm{OPT}(C) \le k$*.*

For the special case of k-MMSA$_3$, it will be convenient to replace the ordered list of literals feeding into each bottom conjunction by the set of input indices appearing in that gate. Specifically, a k-MMSA$_3$ instance can be written as

$$C(x) = \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n} \bigwedge_{t \in N_{i,j}} x_t,$$

where $N_{i,j} \subseteq [n]$ for every $i, j \in [n]$.

The following lemma shows that given a k-MMSA$_3$ instance, one can prune any bottom conjunction term with $|N_{i,j}| > k$ without hurting the gap.

▶ **Lemma 2.4.** *There is a polynomial-time deterministic algorithm which, given as input a* k*-MMSA*$_3$ *instance*

$$C(x) = \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n} \bigwedge_{t \in N_{i,j}} x_t,$$

outputs a k*-MMSA*$_3$ *instance*

$$C'(x) = \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n} \bigwedge_{t \in N'_{i,j}} x_t,$$

where for every $i \in [n]$ *and* $j \in S_i$*,* $|N_{i,j}| \le k$*. Furthermore,*

- *If* $\mathrm{OPT}(C) \le k$*, then* $\mathrm{OPT}(C') \le k$*.*
- $\mathrm{SAT}(C') \subseteq \mathrm{SAT}(C)$*. In particular, for every* $h > k$*, if* $\mathrm{OPT}(C) > h$*, then* $\mathrm{OPT}(C') > h$*.*

Proof. C' is obtained from C by simply removing every bottom conjunction with $|N_{i,j}| > k$. That is, let $S_i := \{j \in [n] \mid |N_{i,j}| \le k\}$, and let

$$C'(x) := \bigwedge_{i=1}^{n} \bigvee_{j \in S_i} \bigwedge_{t \in N_{i,j}} x_t.$$

To pad the size of the second layer to n, we simply duplicate its inputs.

Obviously, $C(x) \ge C'(x)$, hence $\mathrm{SAT}(C') \subseteq \mathrm{SAT}(C)$. On the other hand, suppose $\mathrm{OPT}(C) \le k$, and let x be a satisfying assignment to C with $\|x\|_0 \le k$. Since $|N_{i,j}| > k$ for every $i \in [n]$ and $j \notin S_i$, we have $\bigwedge_{t \in N_{i,j}} x_t = 0$, thus

$$\begin{aligned} C'(x) &= \bigwedge_{i=1}^{n} \left(\left(\bigvee_{j \in S_i} \bigwedge_{t \in N_{i,j}} x_t \right) \vee \left(\bigvee_{j \notin S_i} \bigwedge_{t \in N_{i,j}} x_t \right) \right) \\ &= \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n} \bigwedge_{t \in N_{i,j}} x_t = C(x). \end{aligned}$$

Hence x is also a satisfying assignment to C', which implies that $\mathrm{OPT}(C') \le k$. ◀

We next introduce two closely related parameterized problems: k-MaxCover and k-MinLabel.

▶ **Definition 2.5** (k-MaxCover and k-MinLabel)**.** *k-MaxCover and k-MinLabel are both defined on a partitioned bipartite graph $G = (A, B, E)$ where $A = A_1 \dot\cup \ldots \dot\cup A_k, B = B_1 \dot\cup \ldots \dot\cup B_m$ and $m \le n$, with different optimizing objectives.*

We say a set $X \subseteq A$ (possibly contain multiple vertices from the same part) covers a right part B_j if there exists some $b \in B_j$, such that for every $i \in [k]$, there exists a vertex $a_i \in A_i \cap X$ which is connected to b.

The goal of k-MaxCover is to pick one vertex a_i from each left part A_i, to cover the maximum fraction of right parts B_j, while the goal of k-MinLabel is to pick minimum-sized subset $X \subseteq A$, to cover every right part B_j. We write the two values as $\mathsf{MaxCover}(G)$ *and* $\mathsf{MinLabel}(G)$*, respectively. Specifically,*

$$\mathsf{MaxCover}(G) = \frac{1}{m}\left(\max_{a_1\in A_1,\ldots,a_k\in A_k} \left|\left\{j \in [m] \;\middle|\; \exists b \in B_j \text{ such that } \forall i \in [k],\ (a_i, b) \in E\right\}\right|\right),$$

$$\mathsf{MinLabel}(G) = \min_{X\subseteq A, X \text{ covers every } B_j,\ j\in[m]} |X|.$$

There is a canonical reduction from $(1, \varepsilon)$-gap k-MaxCover to $(1, \varepsilon^{-1/k})$-gap k-MinLabel, as formalized in [4, 17]. We omit the proof here and refer the reader to [17, Proposition A.1].

▶ **Lemma 2.6** ([4, 17])**.** *Let $G = (A, B, E)$ be a k-MaxCover/k-MinLabel instance with $A = A_1 \dot\cup \cdots \dot\cup A_k$ and $B = B_1 \dot\cup \cdots \dot\cup B_m$.*

- *If* $\mathsf{MaxCover}(\Gamma) = 1$*, then* $\mathsf{MinLabel}(\Gamma) = k$.
- *If* $\mathsf{MaxCover}(\Gamma) \le \varepsilon$ *for some* $\varepsilon > 0$*, then* $\mathsf{MinLabel}(\Gamma) \ge (1/\varepsilon)^{1/k} \cdot k$.

We remark that k-MinLabel can be written as a restricted special case of MMSA_4. Therefore, the inapproxiambility of k-MinLabel applies to k-MMSA_4 as well. This observation is formalized in the following lemma.

▶ **Lemma 2.7.** *Let $G = (A, B, E)$ be a k-MinLabel instance with $A = A_1 \dot\cup \cdots \dot\cup A_k$ and $B = B_1 \dot\cup \cdots \dot\cup B_m$. There is a polynomial-time construction of a k-MMSA_4 circuit C_G such that*

$$\mathrm{OPT}(C_G) = \mathsf{MinLabel}(G).$$

Proof. For every left vertex $a \in A$, introduce one Boolean variable x_a. For a right vertex $b \in B$, write

$$N_G(b) = \{a \in A \mid (a, b) \in E\}.$$

Define the monotone circuit

$$C_G(x) = \bigwedge_{j\in[m]} \bigvee_{b\in B_j} \bigwedge_{i\in[k]} \bigvee_{a\in A_i\cap N_G(b)} x_a.$$

After padding dummy gates/variables to make the width of each gate equal to the number of variables, $C_G(x)$ becomes a k-MMSA_4 circuit and can be constructed in polynomial time. It's also easy to check the one-to-one correspondence between a subset $X \subseteq A$ covering every B_j in G and a satisfying assignment of C_G. ◀

We next present a canonical padding lemma which boosts any FPT-time inapproximability for k-MMSA with factor $f(n)^{1/g(k)}$ to $f(n)^{o(1)}$.

▶ **Lemma 2.8** (Boosting the inapproximability of k-MMSA$_t$ by padding). *Fix $t \in \mathbb{N}^+$. Let $f : \mathbb{N} \to \mathbb{R}^+$ be a computable, nondecreasing, unbounded, and padding-stable function in the following sense: for every constant $a \in \mathbb{N}$, there is a sufficiently large $n_0(a)$ such that*

$$f(n+a) \le f(n)^2 \qquad \forall n \ge n_0(a).$$

Let $g : \mathbb{N} \to \mathbb{R}^+$ be a computable, nondecreasing and unbounded function. Suppose for k-MMSA$_t$, no FPT algorithm can distinguish between

$$\mathrm{OPT}(C) \le k \qquad \text{and} \qquad \mathrm{OPT}(C) > k \cdot f(n)^{1/g(k)}.$$

Then, for every computable function $\alpha : \mathbb{N} \to \mathbb{R}^+$ with $\alpha(k) = o(1)$, no FPT algorithm can distinguish between

$$\mathrm{OPT}(C) \le k \qquad \text{and} \qquad \mathrm{OPT}(C) > k \cdot f(n)^{\alpha(k)}.$$

In other words, k-MMSA$_t$ doesn't admit an FPT-time $f(n)^{o(1)}$-approximation.

Proof. Fix $\alpha(k) = o(1)$ and choose K sufficiently large such that $\alpha(K) \le 1/(4g(k))$. Given an MMSA$_t$ circuit C on n variables, add $K - k$ fresh variables which are forced to be one in any satisfying assignment:

$$\widehat{C}(x, y) = C(x) \wedge y_1 \wedge \cdots \wedge y_{K-k}.$$

By representing each new variable as a dummy length-t unary path, the new instance $\widehat{C}$ is still a MMSA$_t$ instance with new parameter K. Furthermore, $\widehat{C}$ has $N = n + K - k$ many variables, satisfies

$$\mathrm{OPT}(\widehat{C}) = \mathrm{OPT}(C) + K - k,$$

and can be constructed in FPT time. Suppose for contradiction that there is an FPT-time algorithm which can distinguish between

$$\mathrm{OPT}(\widehat{C}) \le K \qquad \text{and} \qquad \mathrm{OPT}(\widehat{C}) > K \cdot f(N)^{\alpha(K)}.$$

Since for sufficiently large N depending only on K, padding-stability gives $f(N) = f(n + K - k) \le f(n)^2$, and f is unbounded, we have

$$K \cdot f(N)^{\alpha(K)} \le K \cdot f(n)^{2\alpha(K)} \le K \cdot f(n)^{1/(2g(k))} \le K - k + k \cdot f(n)^{1/g(k)}.$$

Thus, the same algorithm can also distinguish between

$$\mathrm{OPT}(C) \le k \qquad \text{and} \qquad \mathrm{OPT}(C) > k \cdot f(n)^{1/g(k)}$$

in FPT time, a contradiction. ◀

3 An FPT-time $O(2^k \log n)$-Approximation for k-MMSA$_3$

▶ **Theorem 3.1.** *There is a deterministic FPT-time $O(2^k \log n)$-approximation algorithm for k-MMSA$_3$ on n input bits.*

Proof. Write the input instance as

$$C(x) = \bigwedge_{i=1}^{n} \bigvee_{j=1}^{n} \bigwedge_{t \in N_{i,j}} x_t,$$

where $N_{i,j} \subseteq [n]$ for every $i, j \in [n]$. By Lemma 2.4, we can without loss of generality assume that $|N_{i,j}| \le k$.

The algorithm proceeds by repeatedly assigning some variables to 1, until the whole instance C is satisfied. Let

$$S = \{i \in [n] \mid x_i = 1\}$$

be the set of variables that are assigned to 1, the i-th clause

$$\left(\bigvee_{j=1}^{n} \bigwedge_{t \in N_{i,j}} x_t \right)$$

is covered if and only if there exists some term $j \in [n]$ such that $N_{i,j} \subseteq S$. Let

$$\mathrm{Cov}(S) = \{i \in [n] \mid \exists j \in [n] \text{ with } N_{i,j} \subseteq S\}.$$

The algorithm starts with $S_0 = \emptyset$. In the r-th iteration, it takes the set $N_{a,b}$ maximizing the number of newly covered indices

$$\left|\mathrm{Cov}(S_{r-1} \cup N_{a,b}) \setminus \mathrm{Cov}(S_{r-1})\right|,$$

and sets $S_r = S_{r-1} \cup N_{a,b}$. Given the fact that $|N_{i,j}| \leq k$ for every $i, j \in [n]$, it suffices to prove that the algorithm stops after $O(2^k \log n)$ iterations, and thus outputs a satisfying assignment of C with weight at most $O(k \cdot 2^k \log n)$.

Let $x \in \{0,1\}^n$ be a satisfying assignment of C and let $X = \mathrm{supp}(x)$. Fix the r-th iteration, let

$$U_r = [n] \setminus \mathrm{Cov}(S_r)$$

be the set of currently uncovered indices. For every $i \in U_r$, since x satisfies C, there exists some witness $j = j(i) \in [n]$ with $N_{i,j(i)} \subseteq X$. Every chosen witness is a subset of X, and X has at most 2^k distinct subsets. Therefore, by an averaging argument, some set $W \subseteq X$ appears as $N_{i,j(i)}$ for at least $|U_r|/2^k$ indices $i \in U_r$. The best enumerated set covers at least $|U_r|/2^k$ currently uncovered indices. Consequently,

$$|U_{r+1}| \leq \left(1 - \frac{1}{2^k}\right) |U_r|.$$

Iterating this inequality, for $T = \lceil 2^k \ln n \rceil$,

$$|U_T| \leq n \left(1 - \frac{1}{2^k}\right)^T \leq n \exp\left(-\frac{T}{2^k}\right) < 1.$$

Since $|U_T|$ is an integer, $U_T = \emptyset$. Thus, the algorithm returns a satisfying assignment for C with weight at most $kT = O(k2^k \log n)$.

The algorithm is deterministic and runs in FPT time: it performs $O(2^k \log n)$ iterations, and in each iteration it enumerates at most n^2 candidate sets and computes their gains in polynomial time. This proves the theorem. ◀

4 A Reduction from (k, h)-gap MMSA$_3$ to $(2^k, h/k)$-gap MMSA$_2$

▶ **Theorem 4.1.** *There is an FPT-time reduction, which takes as input a k-MMSA$_3$ instance*

$$F(z) = \bigwedge_{i \in [n]} \bigvee_{j \in [n]} \bigwedge_{t \in N_{i,j}} z_t,$$

on $z \in \{0,1\}^n$, outputs a 2^k-MMSA$_2$ instance G on $x \in \{0,1\}^{n^2}$, such that

- *(Completeness) If $\mathrm{OPT}(F) \leq k$, then $\mathrm{OPT}(G) \leq 2^k$.*
- *(Soundness) Fix any $h > k$, if $\mathrm{OPT}(F) > h$, then $\mathrm{OPT}(G) > h/k$.*

The reduction contracts each distinct bottom conjunction of F into one variable of the k-MMSA$_2$ instance G. Namely, for every distinct set $N_{i,j}$ we create a variable intended to mean that all original variables in $N_{i,j}$ are set to 1. Since there are at most n^2 bottom conjunctions, this gives at most n^2 variables.

For completeness, suppose F has a satisfying assignment z whose support Z has size at most k. Every bottom conjunction that is true under z corresponds to a set $N_{i,j} \subseteq Z$ of size $\leq k$, and there are at most 2^k many such sets. Setting the variables representing these true conjunctions to one gives a satisfying assignment of G of weight at most 2^k.

For soundness, suppose G has a satisfying assignment x of weight at most h/k. Decode x back to an assignment z for F by taking the union of all sets represented by the ones in x. Since every represented set has size at most k, the decoded assignment has weight at most h.

Proof of Theorem 4.1. Let

$$\mathcal{S} := \{N_{i,j} \mid i, j \in [n]\}.$$

By Lemma 2.4, we can without loss of generality assume that every set in $\mathcal{S}$ has size at most k. Since $|\mathcal{S}| \leq n^2$, fix an injective map $f : \mathcal{S} \to [n^2]$. For every $i, j \in [n]$, write

$$s_{i,j} := f(N_{i,j}).$$

For every $s \in \mathrm{Im}(f)$, let T_s denote the unique set in $\mathcal{S}$ such that $f(T_s) = s$.

Define the k-MMSA$_2$ instance

$$G(x) := \bigwedge_{i \in [n]} \bigvee_{j \in [n]} x_{s_{i,j}}.$$

The instance uses at most n^2 variables and has size $n^{O(1)}$. It can clearly be constructed in polynomial time, and hence in FPT time.

Completeness. Suppose there exists $z \in \{0,1\}^n$ such that $F(z) = 1$ and $\|z\|_0 \leq k$. Let

$$Z := \mathrm{supp}(z) = \{t \in [n] : z_t = 1\}.$$

Define $x \in \{0,1\}^{n^2}$ by

$$x_s := \begin{cases} 1, & \text{if } s \in \mathrm{Im}(f) \text{ and } T_s \subseteq Z, \\ 0, & \text{otherwise.} \end{cases}$$

We first show that $G(x) = 1$. Fix any $i \in [n]$. Since $F(z) = 1$, there exists some $j \in [n]$ such that

$$\bigwedge_{t \in N_{i,j}} z_t = 1,$$

equivalently, $N_{i,j} \subseteq Z$. As $T_{s_{i,j}} = N_{i,j}$, the definition of x gives $x_{s_{i,j}} = 1$. Therefore every outer clause of G is satisfied, and hence $G(x) = 1$.

It remains to bound the weight of x. If $x_s = 1$, then $s \in \mathrm{Im}(f)$ and $T_s \subseteq Z$. Because f is injective, distinct coordinates correspond to distinct subsets of Z. Hence

$$\|x\|_0 \leq 2^{|Z|} \leq 2^k.$$

Thus $\mathrm{OPT}(G) \leq 2^k$.

Soundness. Fix $h > k$, and suppose for contradiction that there exists $x \in \{0,1\}^{n^2}$ such that $G(x) = 1$ and $\|x\|_0 \leq h/k$. Let

$$\mathcal{A} := \{T_s \mid s \in \mathrm{Im}(f) \text{ and } x_s = 1\}$$

be the family of original bottom sets represented by the selected variables of G, and let

$$U := \bigcup_{T \in \mathcal{A}} T.$$

Define $z \in \{0,1\}^n$ to be the indicator vector of U.

Since every $T \in \mathcal{A}$ has size at most k, and since $|\mathcal{A}| \leq \|x\|_0$, we have

$$\|z\|_0 = |U| \leq \sum_{T \in \mathcal{A}} |T| \leq k \cdot |\mathcal{A}| \leq k\|x\|_0 \leq h.$$

We now show that $F(z) = 1$. Fix any $i \in [n]$. Since $G(x) = 1$, there exists some $j \in [n]$ such that $x_{s_{i,j}} = 1$. By construction, $T_{s_{i,j}} = N_{i,j}$, so $N_{i,j} \in \mathcal{A}$ and therefore $N_{i,j} \subseteq U = \mathrm{supp}(z)$. Hence

$$\bigwedge_{t \in N_{i,j}} z_t = 1.$$

This holds for every $i \in [n]$, so $F(z) = 1$, contradicting $\mathrm{OPT}(F) > h$. ◀

▶ **Corollary 4.2.** *For any computable function $t(n,k)$, if there is an FPT-time $t(n,k)$-approximation for k-MMSA$_2$, then there is an FPT-time $(t(n^2, 2^k) \cdot 2^k)$-approximation for k-MMSA$_3$. Equivalently, if there is no FPT-time $t(n,k)$-approximation for k-MMSA$_3$, then there is no FPT-time $\frac{t(\sqrt{n}, \log k)}{k}$-approximation for k-MMSA$_2$.*

In particular, a $(\log n)^{\Omega(1)}$-factor FPT inapproximability for k-MMSA$_3$ under W[1] $\neq$ FPT, or an $\omega(2^k)$-factor FPT inapproximability for k-MMSA$_3$ under W[2] $\neq$ FPT, could imply new FPT inapproximability results for k-MMSA$_2$ under the respective assumptions, improving [17, 19, 20].

5 A Reduction from k-MMSA$_t$ to Gap k-MMSA$_{t+2}$

In this section, we first summarize known inapproximability results for k-MMSA$_4$, then introduce [22]'s reduction from k-MMSA$_t$ to k-MMSA$_{t+2}$, and finally plug a better set of parameters into it to get improved inapproximability of k-MMSA$_4$ under W[2] $\neq$ FPT and ETH.

5.1 Inapproximability of k-MMSA$_4$ from Literature

The following inapproximability on k-MaxCover was from [17, 18]. We remark that [17] also provides the inapproximability under other assumptions such as SETH or the k-SUM Hypothesis. However, we choose to present only the most relevant assumptions here.

▶ **Lemma 5.1** (Inapproximability of k-MaxCover [17, 18])**.**

- *Assuming* W[1] $\neq$ FPT*, no FPT algorithm can approximate k-MaxCover within an $n^{1/\mathsf{poly}(k)}$ factor.*
- *Assuming* ETH*, no $n^{o(k)}$-time algorithm can approximate k-MaxCover within an $n^{1/\mathsf{poly}(k)}$ factor.*

Passing Lemma 5.1 through the reduction chain of Lemmas 2.6–2.8, we have:

▶ **Corollary 5.2** (Inapproximability of k-MMSA$_4$, implicit in [17, 18]).

- *Assuming* W[1] $\neq$ FPT*, no FPT algorithm can approximate k-MMSA$_4$ within an $n^{o(1)}$ factor.*
- *Assuming* ETH*, no $n^{o(k)}$-time algorithm can approximate k-MMSA$_4$ within an $n^{1/\mathsf{poly}(k)}$ factor.*

Marx [22] also showed a total-FPT-inapproximability for k-MMSA$_4$ under the weaker assumption W[2] $\neq$ FPT.

▶ **Lemma 5.3** ([22]). *Assuming* W[2] $\neq$ FPT*, for any computable function f, no FPT algorithm can approximate k-MMSA$_4$ within an $f(k)$ factor.*

5.2 Revisiting Marx's Reduction

▶ **Theorem 5.4** (Adapted from [22]). *Let $t \in \mathbb{N}^+$ and let $h \geq k$. Given a k-MMSA$_t$ instance C on n variables and an (n, m, Σ, h)-perfect hash family $\mathcal{F} = \{f_1, \ldots, f_m\}$ with $|\Sigma| \geq k$, there is a reduction that outputs a k-MMSA$_{t+2}$ instance C' on the same n variables such that:*

- *the size and running time are at most $\left(m \cdot \binom{|\Sigma|}{k} \cdot |C|\right)^{O(1)}$;*
- *if C has a satisfying assignment of weight at most k, then the same assignment satisfies C';*
- *if every satisfying assignment of C has weight at least $k+1$, then every satisfying assignment of C' has weight more than h.*

Proof. For every $i \in [m]$ and every $T \in \binom{\Sigma}{k}$, write

$$N_{i,T} := f_i^{-1}(T).$$

Define

$$C'(x) := \bigwedge_{i \in [m]} \bigvee_{T \in \binom{\Sigma}{k}} C\big(x \wedge \mathbf{1}_{N_{i,T}}\big).$$

This is a monotone circuit with two additional outer layers, hence an MMSA$_{s+2}$ instance, and its size is bounded by $\left(m \cdot \binom{|\Sigma|}{k} \cdot |C|\right)^{O(1)}$.

For completeness, let x be a satisfying assignment of C with support $X \subseteq [n]$ and $|X| \leq k$. For every $i \in [m]$, choose $T_i \in \binom{\Sigma}{k}$ such that

$$f_i(X) \subseteq T_i.$$

This is possible because $|f_i(X)| \leq |X| \leq k$ and $|\Sigma| \geq k$. Then $X \subseteq N_{i,T_i}$, so

$$x \wedge \mathbf{1}_{N_{i,T_i}} = x.$$

Therefore every outer conjunction of C' is satisfied, and hence $C'(x) = 1$.

For soundness, let x be any satisfying assignment of C' and let $X = \mathrm{supp}(x)$. Since $C'(x) = 1$, for every $i \in [m]$ there exists some $T_i \in \binom{\Sigma}{k}$ such that

$$C\big(x \wedge \mathbf{1}_{N_{i,T_i}}\big) = 1.$$

By the soundness assumption on C, the assignment $x \wedge \mathbf{1}_{N_{i,T_i}}$ must have weight at least $k+1$. Equivalently,

$$|X \cap N_{i,T_i}| \geq k+1 \qquad \text{for every } i \in [m].$$

Suppose toward contradiction that $|X| \leq h$. Since $\mathcal{F}$ is an (n, m, Σ, h)-perfect hash family, there exists some $i^\star \in [m]$ such that $f_{i^\star}$ is injective on X. For this $i^\star$, every $T \in \binom{\Sigma}{k}$ can contain images of at most k elements of X. Thus

$$|X \cap f_{i^\star}^{-1}(T)| \leq k \qquad \forall T \in \binom{\Sigma}{k},$$

contradicting

$$|X \cap N_{i^\star,T_{i^\star}}| \geq k+1.$$

Hence $|X| > h$. ◀

▶ Remark 5.5. We remark that the goal of Marx [22] was to rule out $f(k)$-factor FPT approximation algorithms for any computable function f. Thus, it was sufficient for him to choose h to be larger than the approximation factor $f(k)$. In his hardness proof, h was used as the parameter of an intermediate *FPT cost-approximation algorithm*, which in turn forced h to depend only on k. In contrast, our argument proceeds more directly, allowing us to choose h to be as large as $n^{O(1/k)}$.

We apply Theorem 5.4 using an (n, m, Σ, h)-perfect hash family from Lemma 2.2 with $h = n^{\Theta(1/k)}$, $|\Sigma| = h^2$ and $m = h^{O(1)} \log n$. Then

$$\binom{|\Sigma|}{k} \leq h^{2k},$$

so the reduction size and running time are bounded by

$$\left(m \cdot \binom{|\Sigma|}{k} \cdot |C|\right)^{O(1)} = \left(h^{O(1)} \log n \cdot h^{2k} \cdot |C|\right)^{O(1)} \leq n^{O(1)}.$$

It was known [5] that under ETH, k-MMSA$_2$ (a.k.a. k-SetCover) cannot be solved in $n^{o(k)}$ time. We thus have the following immediate corollary:

▶ **Corollary 5.6.** *Assuming* ETH*, no $n^{o(k)}$-time algorithm can approximate k-MMSA$_4$ within some $n^{O(1/k)}$ factor.*

Using the same set of parameters and invoking the padding lemma (Lemma 2.8), we also have the following W[2]-hardness:

▶ **Corollary 5.7.** *Assuming* W[2] $\neq$ FPT*, no FPT algorithm can approximate k-MMSA$_4$ within any $n^{o(1)}$ factor.*

Corollary 5.6 and Corollary 5.7 improve upon Corollary 5.2 and Lemma 5.3, respectively.

References

1 Michael Alekhnovich, Samuel R. Buss, Shlomo Moran, and Toniann Pitassi. Minimum propositional proof length is np-hard to linearly approximate. *J. Symb. Log.*, 66(1):171–191, 2001. doi:10.2307/2694916.

2 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995. doi:10.1145/210332.210337.

3 Sanjeev Arora and Carsten Lund. Hardness of approximations. In Dorit S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, pages 399–446. PWS Publishing, Boston, MA, 1997.

4 Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From gap-ETH to FPT-inapproximability: Clique, dominating set, and more. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 743–754. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.74.

5 Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. Strong computational lower bounds via parameterized complexity. *Journal of Computer and System Sciences*, 72(8):1346–1367, 2006. doi:10.1016/j.jcss.2006.04.007.

6 Eden Chlamtáč, Yury Makarychev, and Ali Vakilian. Approximating Red-Blue Set Cover and Minimum Monotone Satisfying Assignment. In Nicole Megow and Adam Smith, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2023)*, volume 275 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.APPROX/RANDOM.2023.11.

7 V. Chvátal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, 1979. doi:10.1287/MOOR.4.3.233.

8 I. Dinur and D. Steurer. Analytical approach to parallel repetition. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 624–633, 2014. doi:10.1145/2591796.2591884.

9 Irit Dinur and Shmuel Safra. On the hardness of approximating label-cover. *Inf. Process. Lett.*, 89(5):247–254, 2004. doi:10.1016/J.IPL.2003.11.007.

10 Rodney G. Downey and Michael R. Fellows. Fixed-parameter tractability and completeness I: Basic results. *SIAM J. Comput.*, 24(4):873–921, 1995. doi:10.1137/S0097539792228228.

11 Rodney G Downey and Michael R Fellows. *Fundamentals of parameterized complexity*, volume 4. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

12 Andreas Emil Feldmann, Karthik C. S., Euiwoong Lee, and Pasin Manurangsi. A survey on approximation in parameterized complexity: Hardness and algorithms. *Algorithms*, 13(6):146, 2020. doi:10.3390/A13060146.

13 Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, Berlin, 2006. doi:10.1007/3-540-29953-X.

14 Michael L. Fredman and János Komlós. On the size of separating systems and families of perfect hash functions. *SIAM Journal on Algebraic Discrete Methods*, 5(1):61–68, 1984. doi:10.1137/0605009.

15 Michael H. Goldwasser and Rajeev Motwani. Intractability of assembly sequencing: Unit disks in the plane. In Frank K. H. A. Dehne, Andrew Rau-Chaplin, Jörg-Rüdiger Sack, and Roberto Tamassia, editors, *Algorithms and Data Structures, 5th International Workshop, WADS '97, Halifax, Nova Scotia, Canada, August 6-8, 1997, Proceedings*, volume 1272 of *Lecture Notes in Computer Science*, pages 307–320. Springer, 1997. doi:10.1007/3-540-63307-3_70.

16 D. S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9(3):256–278, 1974. doi:10.1016/S0022-0000(74)80044-9.

17 Karthik C. S., Bundit Laekhanukit, and Pasin Manurangsi. On the parameterized complexity of approximating dominating set. *J. ACM*, 66(5):33:1–33:38, 2019. doi:10.1145/3325116.

18 Karthik C. S. and Inbal Livni Navon. On hardness of approximation of parameterized set cover and label cover: Threshold graphs from error correcting codes. In Hung Viet Le and Valerie King, editors, *4th Symposium on Simplicity in Algorithms, SOSA 2021, Virtual Conference, January 11-12, 2021*, pages 210–223. SIAM, 2021. doi:10.1137/1.9781611976496.24.

19 Bingkai Lin. A simple gap-producing reduction for the parameterized set cover problem. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 81:1–81:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.81.

20 Bingkai Lin, Xuandi Ren, Yican Sun, and Xiuhan Wang. Constant approximating parameterized k-SETCOVER is W[2]-hard. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 3305–3316. SIAM, 2023. doi:10.1137/1.9781611977554.ch126.

21 L. Lovász. On the ratio of optimal integral and fractional covers. *Discrete Mathematics*, 13(4):383–390, 1975. doi:10.1016/0012-365X(75)90058-8.

22 Dániel Marx. Completely inapproximable monotone and antimonotone parameterized problems. *J. Comput. Syst. Sci.*, 79(1):144–151, 2013. doi:10.1016/J.JCSS.2012.09.001.

23 Christopher Umans. Hardness of approximating sigma$_2^{\mathrm{p}}$ minimization problems. In *40th Annual Symposium on Foundations of Computer Science, FOCS 1999, New York, NY, USA, October 17-18, 1999*, pages 465–474. IEEE Computer Society, 1999. doi:10.1109/SFFCS.1999.814619.

Cycle Cancellation for Submodular Fractional Allocations and Applications

Chandra Chekuri
University of Illinois at Urbana Champaign, IL, USA

Pooja Kulkarni[1]
University of Chicago, IL, USA

Ruta Mehta
University of Illinois at Urbana Champaign, IL, USA

Jan Vondràk
Stanford University, CA, USA

Abstract

We consider discrete allocation problems where m indivisible goods need to be allocated among n agents. When agents' valuation functions are additive, the well-known cycle canceling lemma [39, 47, 45] plays a key role in the design and analysis of rounding algorithms. In this paper, we prove an analogous lemma for the case of submodular valuations. Our algorithm removes cycles in the support graph of a fractional allocation while guaranteeing that each agent's value, measured using the multilinear extension, does not decrease.

We demonstrate applications of the cycle-canceling algorithm, along with other ideas, to obtain new algorithms and results for three well-studied allocation objectives: max-min (Santa Claus problem), Nash social welfare (NSW), and maximin-share (MMS). For the submodular NSW problem, we obtain a $\frac{1}{5}$-approximation; for the MMS problem, we obtain a $\frac{1}{2}(1-1/e)$-approximation through new simple algorithms. For various special cases where the goods are "small" valued or the number of agents is constant, we obtain tight/best-known approximation algorithms. All our results are in the value-oracle model.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Submodular Optimization, Allocation Problems, Fair Division

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.21

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2511.21099

Funding *Chandra Chekuri*: Supported in part by NSF grant CCF-2402667.
Pooja Kulkarni: Supported in part by NSF grant CCF-2334461 and in part by a gift from Adobe to S.Khuller.
Ruta Mehta: Supported in part by NSF grant CCF-2334461.

Acknowledgements We would like to thank László Végh for useful discussions, in particular posing a question about using continuous local search for solving the multilinear relaxation of the Nash social welfare problem.

1 Introduction

Allocation problems are fundamental to both theory and practice. They arise naturally in diverse applications from optimization to fair division, and have been an influential source of techniques in discrete (combinatorial) optimization. In this paper we give a new tool for addressing allocation problems with submodular valuations.

[1] Most of this work was done when the author was a student at UIUC and a postdoc at Northwestern University.



© Chandra Chekuri, Pooja Kulkarni, Ruta Mehta, and Jan Vondràk;
licensed under Creative Commons License CC-BY 4.0

Consider the problem of allocating a set $\mathcal{M}$ of m *indivisible* items to a set $\mathcal{N}$ of n agents with diverse preferences. An allocation can be viewed as a partition of $\mathcal{M}$ into sets $S_1, S_2, \ldots, S_n$ with S_i being the set given to agent i. Each agent i has a valuation function $v_i : 2^{\mathcal{M}} \to \mathbb{R}_+$ where $v_i(S)$ is the value that they obtain for subset $S \subseteq \mathcal{M}$. We will assume v_is to be *monotone submodular*[2], capturing decreasing marginal returns, and normalized so that $v_i(\emptyset) = 0$. Throughout this work, we denote an instance of the allocation problem by $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ and the set of all allocations as $\Pi_{|\mathcal{N}|}(\mathcal{M})$.

Allocation problems with submodular valuations are extensively studied within optimization [43, 20, 25] and fair division [42, 35, 48, 32, 29] communities under various objective functions. Also, see Section 1.1 for more detailed related works. A prominent example is to allocate items to maximize social welfare: find an allocation $S_1, \ldots, S_n$ to max $\sum_{i\in\mathcal{N}} v_i(S_i)$. While this is trivial for additive valuations (simply assign each good to the agent that values it the most), it is an interesting optimization problem for submodular valuations, and is NP-Hard. A simple greedy algorithm achieves an approximation ratio of 1/2 [44, 38] while a much more sophisticated approach achieves a tight approximation ratio of $(1 - 1/e)$ [49, 15]. The latter result was instrumental in developing the multilinear relaxation approach for submodular maximization problems. In this paper, we are concerned with more challenging objectives arising from fairness, efficiency, and other considerations, with the goal of designing unifying tools and techniques. Three such objectives are: (i) Max-min allocation (also referred to as the (Submodular) Santa Claus problem) [42], (ii) Nash Social Welfare (NSW) [36], and (iii) Maximin-share (MMS) [13]. These problems are well-known to be challenging even when agents' valuation functions are additive, i.e, $v_i(S) = \sum_{j\in S} v_i(\{j\})$ for all $i \in \mathcal{N}$. See Section 1.1 for more details about these problems.

Relax-and-round is a natural approach for most of these problems, where first a good *fractional allocation* is obtained and then rounded to an integral allocation while incurring some loss. The purpose of this paper is to develop a widely applicable technical tool that helps round a fractional allocation effectively under submodular valuations. We further highlight the utility of this tool by instantiating it on the above mentioned three objectives. Our tool is inspired by a corresponding result in the additive setting: A fractional allocation is an $m \cdot n$ dimensional vector $\mathbf{x}$ which we interpret as consisting of values $x_{i,j}$ for $i \in \mathcal{N}$ and $j \in \mathcal{M}$; $x_{i,j} \in [0, 1]$ is the fractional amount of item j that is assigned to agent i. We require that $\sum_i x_{i,j} = 1$ for each item j. We let $\mathbf{x}_i$ denote the m-dimensional vector consisting of the variables $x_{i,j}$, $j \in [m]$. Each fractional allocation $\mathbf{x}$ defines an allocation graph $G_{\mathbf{x}}$ which is a bipartite graph with agents on one side and items on the other side and an edge between i and j labeled with the number $x_{i,j}$ iff $x_{i,j} > 0$; in other words it is a representation of the support of $\mathbf{x}$. For a fractional allocation $\mathbf{x}$ the value of agent i in the additive setting, denoted by $v_i(\mathbf{x}_i)$ is naturally defined as $\sum_j v_i(\{j\})x_{i,j}$. Then the following rounding procedure is known.

▶ **Lemma 1** ([39, 47]). *Let* $\mathbf{x}$ *be a fractional allocation and* v_i *additive functions. Then there is another allocation* $\mathbf{y}$ *in the support of* $\mathbf{x}$ *such that (i) the allocation graph* $G_{\mathbf{y}}$ *has no cycles and (ii) for all agents* i, $v_i(\mathbf{y}_i) \geq v_i(\mathbf{x}_i)$. *Furthermore, the allocation* $\mathbf{y}$ *can be rounded to an integral allocation* $\mathcal{A} = (A_1, \ldots, A_n)$ *such that for all agents* i, $v_i(A_i) \geq V_i(\mathbf{y}_i) - \alpha_i$ *where* $\alpha_i = \max_{j:y_{i,j}>0} v_i(\{j\})$.

[2] A real-valued set function $f : 2^{\mathcal{M}} \to \mathbb{R}_+$ is submodular if $f(A + g) - f(A) \geq f(B + g) - f(B)$ for any $A \subseteq B$ and $g \in B \setminus A$; in other words the function exhibits diminishing marginal utility. Monotonicity means that $f(A) \leq f(B)$ for all $A \subseteq B$.

▶ **Corollary 2.** *Suppose* $\mathbf{x}$ *is a fractional allocation such that* $\max_{j:x_{i,j}>0} v_i(\{j\}) \leq \epsilon v_i(\mathbf{x}_i)$ *for each agent* i*. Then there is an integral allocation* $\mathbf{z}$ *such that* $v_i(\mathbf{z}_i) \geq (1-\epsilon)v_i(\mathbf{x}_i)$*.*

The corollary immediately allows one to find a good approximation in the so-called "small" items scenario. One can view the lemma and the corollary as interpolating between the fully fractional setting (which can be solved optimally via LP) and the discrete setting in which no single item is too important to an agent. The utility of the lemma can be understood by considering an alternative rounding strategy: randomly allocate each item independently to an agent based on fractional solution. The parameters obtained by such a rounding are much weaker – a logarithmic factor in the number of agents is typically lost since one has to rely on concentration inequalities followed by a union bound.

Here we generalize the preceding lemma to the submodular setting. In order to do this we need to work with a continuous extension of submodular functions. As mentioned above, for maximization problems the multilinear extension [14] has played a natural role.

▶ **Definition 3** (Multilinear Extension of f)**.**

$$F(x) \coloneqq \sum_{S \subseteq V} f(S) \prod_{j \in S} x_j \prod_{j \in V \setminus S} (1 - x_j) \tag{1}$$

It inherits some useful properties of submodularity and moreover, its value and partial derivatives can be evaluated for a given point $\mathbf{x}$ via random sampling.

Our first contribution is to formulate and prove the following theorem.

▶ **Theorem 4.** *Given an instance of the allocation problem* $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$ *with monotone submodular valuations* v_i*, and a fractional allocation* $\mathbf{x}$ *of the goods to the agents, there is a fractional allocation* $\mathbf{x}'$ *such that (i) the allocation graph* $G_{\mathbf{x}'}$ *is acyclic, and (ii)* $V_i(\mathbf{x}') \geq V_i(\mathbf{x})$ *for all agents* $i \in \mathcal{N}$*; where* V_i *is the multilinear extension of* v_i*.*

We state the preceding theorem in a non-constructive way. A polynomial-time algorithm requires one to use approximations due to the sampling aspects of the multilinear extension. We defer formal details of this to a full version of the paper since they are somewhat standard. An efficient version of the lemma yields an allocation that loses a $(1 - o(1))$-factor in the value for each agent which we ignore in this version. See the end of Section 2 for more details.

Next we discuss application of this lemma to obtained improved algorithms and/or results for the three objectives mentioned above.

1.1 Applications

We demonstrate the utility of Theorem 4 on three problems that arise in fair allocation. The focus is not on obtaining improved approximation factors, but rather to illustrate how the cycle canceling procedure can be applied in conjunction with other ideas to design new relaxation-based algorithms. Table 1 gives a summary of our results.

All the three applications we discuss in this paper have seen a lot of work in the literature. While the table gives the current best-known guarantees, we discuss more details here.

[3] The results from [43] are for welfare maximization. However, the structure of instance used for the reduction directly implies the same bound for the problems mentioned here.

Table 1 Summary of Applications of Theorem 4. For the result marked with (*), while our result matches the existing result, we need ϵ to be a constant independent of n in this paper whereas the prior work required ϵ to be $\Omega(1/\log n)$. Results marked with (**) are information theoretic lower bounds whereas the other results are NP-hardness.

Fairness Notion	Problem Setting	Lower Bound	Prior Work	This Paper
Max-min	$v_i(j) \leq \epsilon\mathsf{OPT}\ \forall i,j$	–	–	$(1-1/e-\epsilon)$
Max-min	Constant n	$(1-1/e+\epsilon)$ [43] **	$(1-1/e-\epsilon)$ [20]	$(1-1/e-\epsilon)$*
NSW	General	$(1-1/e+\epsilon)$ [32]	$\approx 1/3.9$ [11]	1/5
NSW	Constant n	$(1-1/e+\epsilon)$ [32]	$(1-1/e-\epsilon)$ [32]	$(1-1/e-\epsilon)$*
MMS	General	$(1-1/e+\epsilon)$ [43][3]**	≈ 0.370 [48]	≈ 0.316
MMS	$v_i(j) \leq \epsilon\mathsf{MMS}_i\ \forall i,j$	–	≈ 0.370 [48]	$(1-1/e-\epsilon)$

1.1.1 Santa Claus (Max-min)

In Section 3.1, we investigate the max-min objective where given an instance of the fair allocation problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$, the goal is to output an allocation $\mathcal{A} = \{A_1, \ldots, A_n\} \in \Pi_{|\mathcal{N}|}(\mathcal{M})$ that maximizes $\min_{i\in\mathcal{N}} v_i(A_i)$. It is alternatively referred to as the Santa Claus problem.

1.1.1.1 Related work

Max-min was first considered as a fair allocation problem by [42]. There has been substantial work since then. Despite this, the approximability is still not well understood even in the additive setting. [12] showed that one cannot approximate max-min to a factor better than 2 in the additive setting unless P = NP. On the positive side, [8] gave an $O(\frac{\log\log m}{\log\log\log m})$-approximation algorithm in the restricted assignment setting (i.e., for each agent $v_i(\{j\}) \in \{v_j, 0\}$). The current best approximation for this setting is $\frac{1}{4+\epsilon}$ [22]. In the general setting, [17] gave a $\tilde{\Omega}(\frac{1}{n^\epsilon})$ approximation algorithm in time $n^{O(1/\epsilon)}$; in particular this implies a $\Omega(\frac{1}{poly(\log n)})$-approximation in quasi-polynomial time. These results were extended to the submodular setting: [7] gave a $\Omega(\frac{1}{poly(\log n)})$-approximation algorithm with running time $n^{O(\log n/\log\log n)}$, and for any fixed $\epsilon > 0$; a $\tilde{\Omega}(\frac{1}{n^\epsilon})$-approximation algorithm with running time $n^{O(1/\epsilon)}$. For the restricted case with submodular valuations, [6] gave an $\Omega(\frac{1}{\log\log n})$-approximation algorithm. Here, restricted submodular Santa Claus is defined as follows: there is a common submodular valuation function v, each agent has a subset $X_i \subseteq \mathcal{M}$ and the value of agent i is $v_i(S) = v(S \cap X_i)$. Another result known for additive valuations was by [12] who showed that there is always an allocation which gives each agent a value of at least $(\mathsf{OPT} - \max_{i\in\mathcal{N}, j\in\mathcal{M}} v_i(j))$ where OPT is the optimal max-min value, and this is via the cycle-cancellation lemma.

1.1.1.2 Our Contributions

In this work, we study the *submodular* Santa Claus problem. We show an additive approximation guarantee that is analogous to the result in [12] for additive valuations. While additive and multiplicative approximation guarantees are incomparable in general, when the goods are small valued, additive approximation imply multiplicative approximations. We thereby achieve the first non-trivial multiplicative approximation for the small-goods case with submodular valuations. Furthermore, we give a $(1-1/e-\epsilon)$ approximation for the case when the instance has a fixed number of agents. This was previously known from [20], however, we achieve a more efficient algorithm (see Remark 15).

1.1.2 Nash social welfare (NSW)

In Section 3.2, we investigate Nash social welfare which is a popular fairness *and* efficiency notion. The objective is to find an allocation $\mathcal{A} = (A_1, \ldots, A_n)$ that maximizes the geometric mean of agent valuations i.e., $\mathcal{A} = \text{argmax}_{S \in \Pi_{|\mathcal{N}|}(\mathcal{M})} \left(\prod_{i \in [n]} v_i(S_i)\right)^{1/n}$.

1.1.2.1 Related work

The Nash social welfare (NSW) objective has been proved to offer a natural balance between utilitarian welfare (sum of utilities) and fairness (max–min welfare) [36, 16]. Even for additive valuations, maximizing NSW is APX-hard [37] and obtaining a constant factor approximation was challenging. The first constant factor for additive valuations was presented by Cole and Gkatzelis [24], using a market equilibrium approach. Alternative approaches were found afterwards, using either market equilibria or fractional relaxations, that provide constant factors for additive, budget-additive and piecewise-linear concave valuations [4, 5, 23, 10, 28]. For submodular valuations, [32] gave the first approximation independent of the number of goods using a "matching-rematching" along with greedy allocation approach. [30] extended the matching-rematching approach along with fractional allocations and proved a constant factor for "Rado valuations" (a subclass of submodular valuations related to matroids). This framework was thereafter extended to a 1/380-approximation for submodular valuations by [40], which was subsequently improved to a 1/4-approximation using matching-rematching and local search [29]. A recent line of work [27, 26, 11] studied the *weighted* NSW problem, maximizing the weighted geometric mean of valuations, and obtained constant-factor approximations for additive and submodular valuations even in this setting. For additive valuations, their guarantees match the best known bound for unweighted NSW ($e^{1/e}$) from [10]; for submodular valuations, the most recent approximation factor is $\sim \frac{1}{3.56}$ [11] via computer-aided analysis. These algorithms are based on a configuration-type relaxation and randomized rounding.

1.1.2.2 Our Contributions

Our primary result here is to present a simple algorithm that still achieves a small constant-factor approximation for submodular NSW. Our algorithm is based on the matching-relaxation-rematching framework like [32, 30, 40] and provides a 1/5-approximation guarantee. Additionally, we also give a $(1 - 1/e - \epsilon)$-approximation for the case when the instance has constantly many agents in a more efficient manner than was previously done by [32] (see Remark 21).

1.1.3 Maximin Share (MMS)

In Section 3.3, we look at the Maximin share objective. An agent's MMS value, denoted by MMS_i is the maximum value she can ensure for herself if she divides the goods into n parts and chooses the worst part for herself.

1.1.3.1 Related work

Maximin Share (MMS) was defined as a fairness notion by [13]. [46] proved that MMS-allocations i.e., allocations that give every agent their MMS value need not exist even with additive valuations. Therefore, a series of works that obtain α-MMS allocation, i.e., each agent receives a value of α-MMS have been studied for additive valuations [46, 3, 33, 34, 2, 1]

culminating in a $3/4 + O(1)$-approximation. The algorithms in these works are mostly combinatorial and rely on either matching-based or greedy-based techniques. For general valuations, [9, 35] initiated the study for approximate MMS allocations. [9] gave a $\sim 1/10$ approximation algorithm via round-robin allocation and [35] gave a 1/3-approximation via local search. Recently, [48] gave an algorithm that provides 10/27-approximation to MMS using a greedy algorithm. For a subclass of submodular valuations called SPLC, [19] gave a 1/2 approximation algorithm based on market-equilibrium solution followed by rounding. Several other works on MMS on other valuation functions or in constrained settings exist that we do not list here.

1.1.3.2 Our Contributions

Our main theorem naturally extends the equilibrium-rounding approach to submodular valuations to give a $\frac{1}{2}(1-1/e-o(1))$-approximation algorithm. Additionally, when the goods are small compared to the MMS value (at most ϵMMS), we obtain a novel $(1-1/e-\epsilon)$-MMS allocation using the same approach.

2 Cycle-Cancellation under Multilinear Extension

In this section we present the main technical contribution of this work – Theorem 4. This theorem states that given an instance $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ of the allocation problem and any fractional allocation $\mathbf{x}$, we can convert it to an allocation with at most $|\mathcal{N}|-1$ fractional variables whose allocation graph forms a forest, while retaining the value received by any agent *under the multilinear relaxation*. It is well-known that when the function is submodular, the multilinear extension is convex in any $\mathbf{e}_i - \mathbf{e}_j$ direction [14]. We first generalize this to show that the multilinear extension is convex in any direction given by $t_i\mathbf{e}_i - t_j\mathbf{e}_j$, where $t_i, t_j \geq 0$. This allows us to argue how the function value changes when we move goods along a cycle allowing us to eventually cancel cycles in the allocation graph.

▶ **Theorem 4.** *Given an instance of the allocation problem $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ with monotone submodular valuations v_i, and a fractional allocation $\mathbf{x}$ of the goods to the agents, there is a fractional allocation $\mathbf{x}'$ such that (i) the allocation graph $G_{\mathbf{x}'}$ is acyclic, and (ii) $V_i(\mathbf{x}') \geq V_i(\mathbf{x})$ for all agents $i \in \mathcal{N}$; where V_i is the multilinear extension of v_i.*

Before we prove the theorem, we see what happens to the value of the multilinear extension when for two goods i, j with $i \neq j$, we increase the allocation of good i by t_i and decrease the amount of good j by t_j. We remark that an equivalent lemma appears in an independent work on the knapsack problem with a submodular objective [41].

▶ **Lemma 5.** *Given a submodular function f with multilinear extension F, and allocations $\mathbf{x} = (x_1, \ldots, x_m)$ and $\bar{\mathbf{x}} = (x_1, \ldots, x_i + t_i, \ldots, x_j - t_j, \ldots, x_m)$, for any t_i, t_j such that either both $t_i \geq 0$ and $t_j \geq 0$, or both $t_i \leq 0$ and $t_j \leq 0$, we have,*

$$F(\bar{\mathbf{x}}) - F(\mathbf{x}) \geq t_i \frac{\partial F}{\partial x_i}\Big|_{\mathbf{x}} - t_j \frac{\partial F}{\partial x_j}\Big|_{\mathbf{x}}.$$

Proof. Let $\mathbf{x} = (x_1, \ldots, x_m)$, $\bar{\mathbf{x}} = (x_1, \ldots, x_i + t_i, \ldots, x_j - t_j, \ldots, x_m)$ and $P_{\mathbf{x}}(R) = \prod_{g\in R} x_g \prod_{g\in[m]\setminus R}(1 - x_g)$. Note that $\mathbf{x}$ and $\bar{\mathbf{x}}$ differ in probabilities only on the goods i and j. Therefore, when we expand the multilinear extension, the only places the terms differ are on the probabilities of selecting (or not selecting) i or j. We explicitly separate these terms. Towards this, we denote "$R : i, j$" to indicate all sets that include the elements i and j,

"$R : \tilde{i}, \tilde{j}$" are all sets that do not include the elements i and j. "$R : \tilde{i}, j$" and "$R : i, \tilde{j}$" are defined analogously. Further, we define $P_{\downarrow i,j}(R) = \prod_{g \in R \setminus \{i,j\}} x_g \prod_{g \in [m] \setminus R \setminus \{i,j\}} (1 - x_g)$ i.e., the probability of picking a set R according to a distribution $\mathbf{x}$ except that we ignore the probabilities corresponding to i and j. With this notation, we can write the difference of multilinear extensions as:

$$\begin{aligned}
F(\bar{\mathbf{x}}) - F(\mathbf{x}) &= \sum_R (P_{\bar{\mathbf{x}}}(R) - P_{\mathbf{x}}(R)) f(R) \\
&= \sum_{R:\tilde{i},\tilde{j}} P_{\downarrow i,j}(R)(t_j(1-x_i) - t_i(1-x_j) - t_i t_j) f(R) \\
&\quad + \sum_{R:i,\tilde{j}} P_{\downarrow i,j}(R)(t_i t_j + t_i(1-x_j) + x_i t_j) f(R) \\
&\quad + \sum_{R:\tilde{i},j} P_{\downarrow i,j}(R)(t_i t_j - (1-x_i)t_j - t_i x_j) f(R) \\
&\quad + \sum_{R:i,j} P_{\downarrow i,j}(R)(t_i x_j - x_i t_j - t_i t_j) f(R).
\end{aligned}$$

We rewrite the above equations by combining the terms in $t_i t_j$, t_i and t_j together and we get the following equation

$$\begin{aligned}
F(\bar{\mathbf{x}}) - F(\mathbf{x}) =& t_i t_j \sum_{R:\tilde{i},\tilde{j}} P_{\downarrow i,j}(R)(-f(R) + f(R+i) + f(R+j) - f(R+i+j)) \\
&+ t_i \sum_{R:\tilde{i},\tilde{j}} P_{\downarrow i,j}(R)(-(1-x_j)f(R) + (1-x_j)f(R+i) - x_j f(R+j) + x_j f(R+i+j)) \\
&+ t_j \sum_{R:\tilde{i},\tilde{j}} P_{\downarrow i,j}(R)((1-x_i)f(R) - x_i f(R+i) - x_j f(R+j) - x_i f(R+i+j))
\end{aligned}$$

Let $f(R+i) + f(R+j) - f(R) - f(R+i+j) = \alpha_R$. Note that the coefficient of t_i is $\frac{\partial F}{\partial x_i}\Big|_{\mathbf{x}}$ and that of t_j is $-\frac{\partial F}{\partial x_j}\Big|_{\mathbf{x}}$. Substituting these values, we get

$$F(\bar{\mathbf{x}}) - F(\mathbf{x}) = t_i t_j \sum_{R:\tilde{i},\tilde{j}} P_{\downarrow i,j}(R)\alpha_R + t_i \frac{\partial F}{\partial x_i}\Big|_{\mathbf{x}} - t_j \frac{\partial F}{\partial x_j}\Big|_{\mathbf{x}} \geq t_i \frac{\partial F}{\partial x_i}\Big|_{\mathbf{x}} - t_j \frac{\partial F}{\partial x_j}\Big|_{\mathbf{x}}$$

where the second inequality follows since $\alpha_R \geq 0$ for submodular valuation functions and we assumed t_i, t_j to be non-negative. ◀

▶ Remark 6. We note that it is well-known that when f is a submodular function, its multilinear extension F is convex along the direction $e_i - e_j$ for any $i, j \in [m]$. The above lemma proves a stronger version of this result i.e., the multilinear extension is convex along any $t_i e_i - t_j e_j$ for any $i, j \in [m]$ and $t_i, t_j \geq 0$.

We can now prove Theorem 4. Recall that the instance of allocation problem is denoted by $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$ and we assume that each v_i is submodular. We denote the multilinear extension of v_i by V_i. Note that while we write $V_i(\mathbf{x})$, the value of V_i only depends on the part of $\mathbf{x}$ corresponding to agent i's allocation $\mathbf{x}_i$. Additionally, we use $V'_{i,j}$ to denote the partial derivative of V_i at $\mathbf{x}_i$ with respect to good j.

Proof of Theorem 4. The main idea is that if there is a cycle with fractional values x_{ij}, we can move values along the cycle consistently so that no agent loses in value, and some variable becomes an integer. For any given allocation $\mathbf{x}$, consider a cycle $a_0 - g_0 - a_1 - g_1 -$

$\cdots - a_{\ell-1} - g_{\ell-1} - a_0$ where a_i are the agents and g_i are the goods. To reallocate along the cycle, we need to maintain two constraints: (1) the total fractional allocation of each good remains constant, (2) no agent's value (under the multilinear extension) decreases. To maintain the first constraint, we will change the allocation of good i by δ_i on one edge and $-\delta_i$ on the other adjacent edge. Therefore, a reallocation strategy looks as follows:

$$x'_{a_i g_i} = x_{a_i g_i} + \delta_i$$
$$x'_{a_{i+1} g_i} = x_{a_{i+1} g_i} - \delta_i$$

with index operations modulo ℓ. By Lemma 5, for agent a_i not to lose value, it is sufficient to satisfy:

$$\delta_i V'_{a_i,g_i} - \delta_{i-1} V'_{a_i,g_{i-1}} \geq 0 \tag{2}$$

where, the index operations are modulo ℓ. We get ℓ such inequalities corresponding to the ℓ agents on the cycle. We claim that there is always a non-trivial solution to these inequalities, for any choice of V'_{a_i,g_i}, $V'_{a_i,g_{i-1}}$.

First, if $V'_{a_i,g_i} = 0$ for some agent i, we can set $\delta_{i'} = 0$ for all $i' \neq i$, and choose $\delta_i < 0$ to ensure that the gain of every agent is non-negative. Similarly, if $V'_{a_i,g_{i-1}} = 0$, we can set $\delta_{i'} = 0$ for all $i' \neq i-1$ and choose $\delta_{i-1} > 0$ to ensure that the gain of all agents is non-negative.

Hence, we can assume that $V'_{a_i,g_i} > 0$ and $V'_{a_i,g_{i-1}} > 0$ for all agents on the cycle. In this case, for any value δ_0, we can iteratively find values of $\delta_1, \ldots, \delta_{\ell-1}$ of the same sign such that $\delta_i V'_{a_i,g_i} - \delta_{i-1} V'_{a_i,g_{i-1}} = 0$ for $1 \leq i \leq \ell - 1$. These conditions mandate that $\delta_i = \delta_{i-1} \frac{V'_{a_i,g_{i-1}}}{V'_{a_i,g_i}}$, so we obtain inductively

$$\delta_{\ell-1} = \delta_0 \frac{V'_{a_1,g_0}}{V'_{a_1,g_1}} \frac{V'_{a_2,g_1}}{V'_{a_2,g_2}} \cdots \frac{V'_{a_{\ell-1},g_{\ell-2}}}{V'_{a_{\ell-1},g_{\ell-1}}}.$$

The last remaining condition is that

$$\delta_0 V'_{a_0,g_0} - \delta_{\ell-1} V'_{a_0,g_{\ell-1}} \geq 0$$

which is equivalent to

$$\delta_0 V'_{a_0,g_0} - \delta_0 \frac{V'_{a_1,g_0}}{V'_{a_1,g_1}} \frac{V'_{a_2,g_1}}{V'_{a_2,g_2}} \cdots \frac{V'_{a_{\ell-1},g_{\ell-2}}}{V'_{a_{\ell-1},g_{\ell-1}}} V'_{a_0,g_{\ell-1}} \geq 0.$$

We still have a free choice of $\delta_0 \neq 0$, so we can choose its sign appropriately so that the last condition is satisfied. By construction, this satisfies condition (2) for all agents i.

Denote by $\mathbf{x}'$ the modified fractional solution where $x'_{a_i,g_i} = x_{a_i,g_i} + \delta_i$. By the discussion above, all the δ_i's are determined by δ_0. We want to choose the magnitude of δ_0 so that $\mathbf{x}'$ is feasible and at least one variable on the cycle becomes an integer. This can be accomplished by choosing the supremum or infimum of δ_0 (depending on the desired sign) such that $\mathbf{x}'$ defined by the changes described above is still feasible. We claim that such $\mathbf{x}'$ must have a new integer variable: If all the variables on the cycle are fractional, then there is a larger/smaller value of δ_0 that we can choose and $\mathbf{x}'$ is still feasible. An extreme choice of δ_0 still defines a feasible solution, because the assignment polytope is a closed set.

Finally, we argue that the value for any agent cannot decrease in this operation. After choosing δ_0 as above, the remaining δ_i for $i \in [\ell]$ are chosen as $\delta_i = \delta_{i-1} \frac{V'_{a_i,g_{i-1}}}{V'_{a_i,g_i}}$. Since the partial derivatives are all positive, all δ_is will have the same sign. Therefore, by Lemma 5, we have $V_{a_i}(\mathbf{x}') \geq V_{a_i}(\mathbf{x})$. Note that it is in this step that we use the convexity of the multilinear extension along the direction that we modify our fractional solution. ◀

Note that the proof does not work for non-monotone submodular functions. We could possibly define δ_i to satisfy (2) even for non-monotone submodular functions, but since two variables could be increasing or decreasing at the same time, the relevant multilinear function could be concave rather than convex along the direction of change.

Notes on computational aspects of the procedure: In the discussion above, we ignore the computational issues of computing or estimating the values of $V_i(\mathbf{x})$ and $\frac{\partial V_i}{\partial x_{ij}}$. These issues can be handled by techniques similar to prior work using the multilinear extension: The values of $V_i(\mathbf{x})$ and its partial derivatives are expectations over a certain product distribution and hence can be estimated by random sampling. The sampling error can be made smaller than $\frac{\alpha_i}{poly(m,n)}$ where $\alpha_i = \max_{j \in \mathcal{M}: x_{ij} > 0} v_i(\{j\})$. Since variables are changing by at most 1 in each step, the true change in $V_i(\mathbf{x})$ might be off by $\frac{\alpha_i}{poly(m,n)}$ compared to our estimates. The number of rounding steps is polynomial in m and n (at least one new variable becomes an integer in each step), and hence we can choose the sampling parameters so that the loss for each agent is less than $\frac{\alpha_i}{poly(m,n)}$ at the end. In many applications, this additive error is negligible compared to the loss of α_i which we shall incur in any case due to the rounding on the remaining forest. For completeness, we give our computational result:

▶ **Theorem 7.** *Given an instance of the allocation problem $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$ with monotone submodular valuations v_i, and a fractional allocation $\mathbf{x}$ of the goods to the agents, in randomized polynomial time, we can compute with high probability another fractional allocation $\mathbf{y}$ such that the support of $\mathbf{y}$ is acyclic, and $V_i(\mathbf{y}) \geq V_i(\mathbf{x}) - \frac{\alpha_i}{poly(m,n)}$ for all agents $i \in [n]$ where V_i is the multilinear extension of v_i and $\alpha_i = \max_{j \in \mathcal{M}: x_{ij} > 0} v_i(\{j\})$.*

3 Applications

In this section we discuss applications of the cycle-cancellation. All applications we discuss use the same rounding based on cycle-cancellation which we call NonUniformPipageRounding and is outlined in Algorithm 1. Similar rounding has appeared in the context where valuations are additive or SPLC[4] (eg., [24, 19]).

Algorithm 1 NonUniformPipageRounding$((\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}}), \mathbf{x})$.

1. Create $\mathbf{x}'$ from $\mathbf{x}$ using Theorem 4.
2. In the forest of $G_{\mathbf{x}'}$, root each tree at an arbitrary agent
3. Allocate each fractional good to its parent in the rooted tree

We can prove the following guarantee of this algorithm.

▶ **Lemma 8.** *Given an instance of the allocation problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$ and any fractional allocation $\mathbf{x}$, Algorithm 1 (NonUniformPipageRounding) finds an allocation $\mathcal{A} = (A_1, \ldots, A_{|\mathcal{N}|}) \in \Pi_{|\mathcal{N}|}(\mathcal{M})$ such that $v_i(A_i) \geq V_i(\mathbf{x}) - v_i(\ell_i)$ for all $i \in \mathcal{N}$, where $\ell_i = max_{j \in \mathcal{M}} v_i(\{j\})$.*

Proof. From Theorem 4, the agents lose no value in converting to $\mathbf{x}'$. Then, since the graph is acyclic, step 2 can be executed. Since each agent in the rounding can lost at most one good (the parent good), the agents lose a single good. By submodularity (in fact, subadditivity), the lemma follows. ◀

[4] SPLC is a class of functions that capture additive valuations and are a subset of submodular valuations.

▶ Remark 9. As mentioned at the end of previous section, we ignore the computational aspects of cycle-cancellation procedure. All our applications use the NonUniformPipageRounding where the rounding incurs loss of one good. The loss due to computational aspects of cycle-cancellation (result of Theorem 7) is negligible compared to this loss and therefore can be safely ignored. For keeping the notation clean, we do not add that loss here. Similar to the computational version of Theorem 4, we get the following computational version:

▶ **Lemma 10.** *Given an instance of the allocation problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ and any fractional allocation $\mathbf{x}$, Algorithm 1* (NonUniformPipageRounding)*, in randomized polynomial time finds with high probability an allocation $\mathcal{A} = (A_1, \ldots, A_{|\mathcal{N}|}) \in \Pi_{|\mathcal{N}|}(\mathcal{M})$ such that $v_i(A_i) \geq V_i(\mathbf{x}) - v_i(\ell_i) - \frac{\ell_i}{poly(|\mathcal{N}|,|\mathcal{M}|)}$ for all $i \in \mathcal{N}$, where $\ell_i = max_{j\in\mathcal{M}} v_i(\{j\})$.*

3.1 Approximating Submodular Max-min

The max-min problem also known as the Santa Claus problem is defined as: given an instance of the allocation problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$, output an allocation $\mathcal{A} = (A_i, \ldots, A_{|\mathcal{N}|}) \in \Pi_{|\mathcal{N}|}(\mathcal{M})$ to maximize $\min_{i\in\mathcal{N}} v_i(A_i)$. This value is called the max-min value and we denote it as $\mathsf{OPT} = \max_{\mathcal{A}\in\Pi_{|\mathcal{N}|}(\mathcal{M})} \min_{i\in\mathcal{N}} v_i(A_i)$.

As discussed in Section 1.1.1, our primary aim is to provide an additive approximation for the Submodular Santa Claus problem. We show that this directly gives a multiplicative approximation for "small" goods. Additionally, we also show how to use cycle-cancellation to achieve a multiplicative approximation for the special case when the instance has a fixed number of agents. The key to these results is the NonUniformPipageRounding subroutine (Algorithm 1). To use the subroutine, we naturally need a fractional solution. To obtain this fractional solution for the max-min problem, we use the following result by [20, 18].

▶ **Theorem 11** ([20, 18])**.** *Let $f_1, \ldots, f_n$ be monotone non-negative submodular functions over a common ground set N, $B_1, B_2, \ldots, B_n \in \mathbb{R}_+$ and let $Q \in [0,1]^{|N|}$ be a solvable*[5] *polytope. For every fixed $\epsilon > 0$ there is a polynomial time randomized algorithm that either outputs a point $\mathbf{x} \in Q$ such that for all i,*

$$F_i(\mathbf{x}) \geq (1 - 1/e - \epsilon)B_i$$

or correctly decides that there is no point $\mathbf{x}$ in Q such that $F_i(\mathbf{x}) \geq B_i$ for all i.

▶ **Theorem 12.** *Consider an instance, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ of the Submodular Santa Claus problem with the optimal value OPT. Denote the maximum value of any good as $\mathrm{Max} = \max_{i\in\mathcal{N}, j\in\mathcal{N}} v_i(j)$. For any fixed $\epsilon > 0$, one can obtain in randomized polynomial time an allocation $\mathcal{A} = (A_1, \ldots, A_{|\mathcal{N}|})$ such that $v_i(A_i) \geq (1 - 1/e - \epsilon)\mathsf{OPT} - \mathrm{Max}$.*

Proof. First, we obtain a fractional allocation $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_{|\mathcal{N}|})$ such that $V_i(\mathbf{x}_i) \geq (1 - 1/e - \epsilon)\mathsf{OPT}$. This can be done as follows. We guess the value of OPT via binary search. We then define Q as a partition matroid constructed as follows: for each good, create $|\mathcal{N}|$ copies, one corresponding to each agent. All copies of a given good belong to the same part, and there are $|\mathcal{M}|$ such parts, corresponding to each good. An independent set of this matroid can include at most one element from each part. This construction naturally defines an allocation: each selected copy corresponds to a specific agent, and that agent receives

[5] A polytope is called solvable if we can optimize linear functions over the polytope in polynomial time.

the associated good. The agents' valuation functions extend naturally while preserving submodularity: given an independent set of the matroid, the value F_i for agent i is computed by considering only the copies associated with that agent. This construction first appeared in [38]. Assuming that OPT is a valid estimate of a value that each agent can achieve, we use Theorem 11 to find an $\mathbf{x} \in Q$ that achieves $V_i(\mathbf{x}_i) \geq (1 - 1/e - \epsilon)OPT$ for each agent. To round this fractional solution, we apply the NonUniformPipageRounding to $\mathbf{x}$, and obtain an integral allocation $\mathcal{A} = (A_1, \ldots, A_{|\mathcal{N}|})$ such that $v_i(A_i) \geq V_i(\mathbf{x}_i) - v_i(\ell_i)$ where ℓ_i is the largest valued good for agent i. Therefore, we obtain $v_i(A_i) \geq (1 - 1/e - \epsilon)\mathsf{OPT} - \text{Max}$. ◀

As a corollary, we obtain the following result when $\text{Max} \leq \epsilon\mathsf{OPT}$.

▶ **Corollary 13.** *Consider an instance of Submodular Santa Claus with a promise that there is a feasible allocation $(S_1^*, S_2^*, \ldots, S_{\mathcal{N}}^*)$ of value OPT such that $v_i(j) \leq \epsilon\mathsf{OPT}$ for each $i \in \mathcal{N}$ and each $j \in S_i$. In this setting, for any fixed $\epsilon' > 0$, there is an efficient randomized algorithm that with high probability either outputs an allocation $(S_1, S_2, \ldots, S_{|\mathcal{N}|})$ such that $v_i(S_i) \geq (1 - 1/e - \epsilon - \epsilon')\mathsf{OPT}$ or correctly refutes the promise.*

3.1.1 Constant Number of Agents

Here we consider the setting with a fixed number of agents. In the additive setting there is a PTAS for this case. For the submodular setting the following result can be obtained.

▶ **Theorem 14.** *For any fixed $\epsilon > 0$ and any fixed number of agents $|\mathcal{N}| \geq 1$, there is a $(1 - 1/e - \epsilon)$-approximation for the Submodular Santa Claus problem.*

Proof Sketch. We sketch the ideas since the essence of this has already been captured in Theorem 12. Suppose we have guessed OPT. Since $|\mathcal{N}|$ is fixed we can guess all the "large" items for each agent i from some fixed optimum allocation $(S_1^*, \ldots, S_{|\mathcal{N}|}^*)$. More formally, for each agent i we guess a set $A_i \subset S_i^*$ such that for all $j \in S_i^* \setminus A_i$, $v_i(A_i + j) - v_i(A_i) \leq \epsilon\mathsf{OPT}$. It is easy to see that there exists such a set with $|A_i| \leq (1/\epsilon + 1)$. Thus there are $|\mathcal{M}|^{O(|\mathcal{N}|/\epsilon)}$ guesses that the algorithm needs to try and this is polynomial when $|\mathcal{N}|$ is fixed. Once these are guessed we can then find a fractional allocation and round as in the proof of Theorem 12. The one good loss for an agent i in this setting is at most $\epsilon v_i(A_i)$ due to the preprocessing. The running time is dominated by the guessing step which incurs an overhead of $|\mathcal{M}|^{O(|\mathcal{N}|/\epsilon)}$. ◀

▶ Remark 15. One can use a different rounding approach for the second step after the guessing the large items and solving the relaxation. This was the approach that was outlined in [20] and is based on randomly allocating the items according the fractional allocation; more precisely we allocate each item j independently to exactly one agent where the probablity of agent i receiving item j is precisely $x_{i,j}$. Since the fractional items are "small", one can then use concentration properties of submodular functions, to argue that each agent will have their value preserved to within a $(1 - \epsilon)$-factor of their fractional value with sufficiently high probability; one then uses union bound over all agents. However, randomized rounding incurs larger errors than our rounding method. To make randomized rounding work, we need to use the fact that the number of agents is small, and we need $\epsilon = \Omega(1/\log|\mathcal{N}|)$ for the union bound rather than a fixed constant as above.

3.2 Nash social welfare with submodular valuations

Nash Social Welfare (NSW) is another possible objective in allocation problems: Given a set $\mathcal{M}$ of m items, and a set $\mathcal{N}$ of n agents with valuations $(v_i : i \in \mathcal{N})$, find an allocation $\mathcal{A} = (A_1, \ldots, A_n)$ to maximize the Nash social welfare objective,

$$\left(\prod_{i \in \mathcal{N}} v_i(A_i)\right)^{1/n}.$$

In this section, we present a new $\frac{1}{5}$-approximation algorithm for this problem with monotone submodular valuations, using the rounding lemma.

As mentioned in Section 1.1.2, our algorithm follows the matching-rematching approach that has been used for maximizing NSW with submodular valuations [32, 40, 29]. [40] used this framework with a log-multilinear relaxation and randomized rounding, which gave the first constant factor for NSW with submodular valuation [40]; however, the factor was rather small ($\approx 1/380$). Later, the factor was significantly improved to $1/4$ [29], by replacing the log-multilinear relaxation by discrete local search. Here we return to the log-multilinear relaxation and show that it can be actually used to extract a good approximation factor as well. This is accomplished primarily by the new rounding lemma. In addition, we provide an alternative way to solve the log-multilinear relaxation (approximately) by continuous local search; this also helps in improving the approximation factor, but the effect of this improvement is relatively minor (essentially replacing $1/e$ by $1/2$).

3.2.1 A $\frac{1}{5}$-Approximation Algorithm

Let us describe our new algorithm. It consists of 3 phases: (1) initial matching, (2) relaxation, (3) rematching. The relaxation phase is the new contribution here: we solve the log-multilinear relaxation using continuous local search, and then round it using the non-uniform rounding lemma. A formal description of the algorithm follows.

Algorithm 2 1/5-approximation for NSW with submodular valuations.

Input : Instance $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$
Output : An allocation of items with NSW at least $\frac{1}{5}$OPT

$\tau :=$ matching $\mathcal{N} \to \mathcal{M}$ maximizing $\sum_{i \in \mathcal{N}} \log v_i(\tau(i))$
$H := \tau[\mathcal{N}], \mathcal{M}' = \mathcal{M} \setminus H$
$\mathbf{x} :=$ ContinuousLocalSearch$(\mathcal{M}', \{v_i : i \in \mathcal{N}\})$
$(S_1, \ldots, S_n) :=$ NonUniformPipageRound$(\mathbf{x}, \{v_i : i \in \mathcal{N}\})$
$\sigma :=$ matching $\mathcal{N} \to \mathcal{M}$ maximizing $\sum_{i \in \mathcal{N}} \log v_i(S_i + \sigma(i))$
return $(S_i + \sigma(i) : i \in \mathcal{N})$

First, we have the following simple lemma about the initial matching.

▶ **Lemma 16.** *If $\tau : \mathcal{N} \to \mathcal{M}$ is the initial optimal matching, then for every agent $i \in \mathcal{N}$ and every remaining item $j \in \mathcal{M} \setminus \tau[N]$, $v_i(j) \le v_i(\tau(i))$.*

Proof. If $v_i(j) > v_i(\tau(i))$, then we could improve the matching by redefining $\tau(i) := j$; note that this item is not allocated to anybody else in the matching. ◀

Algorithm 3 ContinuousLocalSearch$(\mathcal{M}', (v_i : i \in \mathcal{N})$.

Let $P := \{\mathbf{x} \in \mathcal{R}_+^{\mathcal{M}' \times \mathcal{N}} : \forall j \in \mathcal{M}, \sum_{i \in \mathcal{N}} x_{ij} = 1\}$ (the assignment polytope)
Initialize $\mathbf{x}^{(0)} := \frac{1}{n}\mathbf{1} \in P$, and $t := 0$
Define the function $F(\mathbf{x}) = \sum_{i \in [n]} \log V_i(\mathbf{x}_i)$.
while $\exists \mathbf{y} \in P, (\mathbf{y} - \mathbf{x}) \cdot \nabla F(\mathbf{x}) > 0$ **do**
  $\mathbf{x}^{(t+1)} := \mathbf{x}^{(t)} + \epsilon(\mathbf{y} - \mathbf{x}^{(t)})$
  $t = t + 1$

Next, we analyze the continuous local search algorithm. Here we ignore numerical issues of convergence of the algorithm; we assume that the output is a local optimum, which can be implemented up to an arbitrarily small error. Note that here we do not state the computational versions of our results. To implement Algorithm 3 in polynomial time, we need to discretize the process like [21]. Let P denote the assignment polytope as in the description of ContinuousLocalSearch.

▶ **Lemma 17.** *Suppose that* $\mathbf{x} \in P$ *is the output of* ContinuousLocalSearch*, and* $\mathbf{x}^* \in P$ *is any other feasible solution. Then*

$$\sum_{i \in \mathcal{N}} \frac{V_i(\mathbf{x}_i^*)}{V_i(\mathbf{x}_i)} \leq 2|\mathcal{N}|.$$

Proof. If $\mathbf{x}$ is a local optimum, then we have

$$(\mathbf{x}^* - \mathbf{x}) \cdot \nabla F(\mathbf{x}) = \sum_{i \in \mathcal{N}} (\mathbf{x}_i^* - \mathbf{x}_i) \cdot \nabla(\log V_i(\mathbf{x}_i)) \leq 0.$$

Consider the points $\mathbf{x} \vee \mathbf{x}^*$ and $\mathbf{x} \wedge \mathbf{x}^*$ (the coordinate maximum and minimum). We can write $(\mathbf{x} \vee \mathbf{x}^*) + (\mathbf{x} \wedge \mathbf{x}^*) = \mathbf{x} + \mathbf{x}^*$, and hence

$$\sum_{i \in \mathcal{N}} ((\mathbf{x}_i \vee \mathbf{x}_i^*) - \mathbf{x}_i) \cdot \nabla(\log V_i(\mathbf{x}_i)) \leq \sum_{i \in \mathcal{N}} (\mathbf{x}_i - (\mathbf{x}_i \wedge \mathbf{x}_i^*)) \cdot \nabla(\log V_i(\mathbf{x}_i)).$$

By the chain rule, $\nabla(\log V_i(\mathbf{x}_i)) = \frac{\nabla V_i(\mathbf{x}_i)}{V_i(\mathbf{x}_i)}$. Also, by submodularity of V_i, we have $((\mathbf{x}_i \vee \mathbf{x}_i^*) - \mathbf{x}_i) \cdot \nabla V_i(\mathbf{x}_i) \geq V_i(\mathbf{x}_i \vee \mathbf{x}_i^*) - V_i(\mathbf{x}_i)$. Similarly, $(\mathbf{x}_i - (\mathbf{x}_i \wedge \mathbf{x}_i^*)) \cdot \nabla V_i(\mathbf{x}_i) \leq V_i(\mathbf{x}_i) - V_i(\mathbf{x}_i \wedge \mathbf{x}_i^*)$ From here and the inequality above,

$$\sum_{i \in \mathcal{N}} \frac{V_i(\mathbf{x}_i \vee \mathbf{x}_i^*) - V_i(\mathbf{x}_i)}{V_i(\mathbf{x}_i)} \leq \sum_{i \in \mathcal{N}} \frac{V_i(\mathbf{x}_i) - V_i(\mathbf{x}_i \wedge \mathbf{x}_i^*)}{V_i(\mathbf{x}_i)}.$$

This inequality is valid even for non-monotone submodular functions. Since here we are dealing with monotone submodular functions, we can drop $V_i(\mathbf{x}_i \wedge \mathbf{x}_i^*)$ and estimate $V_i(\mathbf{x}_i \vee \mathbf{x}_i^*) \geq V_i(\mathbf{x}_i^*)$, which gives

$$\frac{1}{n} \sum_{i \in \mathcal{N}} \frac{V_i(\mathbf{x}_i^*)}{V_i(\mathbf{x}_i)} \leq 2.$$ ◀

Next, we apply the non-uniform pipage rounding procedure. Given a fractional solution $\mathbf{x}$, it produces an allocation $(S_1, \ldots, S_n)$ of the items in $\mathcal{M}'$ such that

$$v_i(S_i) \geq V_i(\mathbf{x}_i) - v_i(\ell(i))$$

where

$$\ell(i) = \operatorname*{argmax}_{j \in \mathcal{M}'} v_i(j).$$

For the analysis of the last stage, we need the following "rematching lemma", which has appeared in various forms in this line of work. The variant we use here appears in [31].

▶ **Lemma 18.** *If $\tau : \mathcal{N} \to \mathcal{M}$ is an optimal initial matching, $H = \tau[\mathcal{N}]$, $\pi : \mathcal{N} \to H$ is any other matching on H, $(S_i : i \in \mathcal{N})$ is any partition of $\mathcal{M}' = \mathcal{M} \setminus H$, and $\ell(i) = \operatorname{argmax}_{j \in \mathcal{M}'} v_i(j)$, then there is a matching $\sigma : \mathcal{N} \to H$ such that*

$$\prod_{i \in \mathcal{N}} v_i(S_i + \sigma(i)) \geq \prod_{i \in \mathcal{N}} \max\{v_i(S_i), v_i(\ell(i)), v_i(\pi(i))\}.$$

Using the rematching lemma, we conclude the analysis as follows.

▶ **Theorem 19.** *If* OPT *is the* NSW *value of the optimal solution, then our algorithm finds a solution of value*

$$\left(\prod_{i \in \mathcal{N}} v_i(S_i + \sigma(i))\right)^{1/n} \geq \frac{1}{5}\mathsf{OPT}.$$

Proof. Let the optimal allocation be $(H_i \cup O_i : i \in \mathcal{N})$, where $H_i \subset H$ and $O_i \subset \mathcal{M}' = \mathcal{M} \setminus H$. Let $\pi : \mathcal{N} \to H$ be a matching such that $\pi(i) = \operatorname{argmax}_{j \in H_i} v_i(j)$ if $H_i \neq \emptyset$, and otherwise $\pi(i)$ an arbitrary item in H so that π is matching. By submodularity, we have

$$v_i(H_i \cup O_i) \leq v_i(O_i) + |H_i| v_i(\pi(i)) = V_i(\mathbf{x}_i^*) + |H_i| v_i(\pi(i))$$

where $\mathbf{x}_i^* = \mathbf{1}_{O_i}$.

From Lemma 18, we have that the value obtained by our algorithm is

$$\mathsf{ALG} = \left(\prod_{i \in \mathcal{N}} v_i(S_i + \sigma(i))\right)^{1/n} \geq \left(\prod_{i \in \mathcal{N}} W_i\right)^{1/n}$$

where $W_i = \max\{v_i(S_i), v_i(\ell(i)), v_i(\pi(i))\}$. From the pipage rounding procedure, we get $v_i(S_i) \geq V_i(\mathbf{x}_i) - v_i(\ell(i))$. Hence,

$$W_i \geq \max\{V_i(\mathbf{x}_i) - v_i(\ell(i)), v_i(\ell(i)), v_i(\pi(i)\} \geq \max\{\frac{1}{2}V_i(\mathbf{x}_i), v_i(\pi(i))\}.$$

Finally, we estimate by AM-GM

$$\begin{aligned} \frac{\mathsf{OPT}}{\mathsf{ALG}} &\leq \left(\prod_{i \in \mathcal{N}} \frac{v_i(H_i \cup O_i)}{W_i}\right)^{1/n} \leq \frac{1}{n}\sum_{i \in \mathcal{N}} \frac{v_i(H_i \cup O_i)}{W_i} \\ &\leq \frac{1}{n}\sum_{i \in \mathcal{N}} \frac{V_i(\mathbf{x}_i^*) + |H_i| v_i(\pi(i))}{W_i} \leq \frac{1}{n}\sum_{i \in \mathcal{N}} \left(\frac{V_i(\mathbf{x}_i^*)}{\frac{1}{2}V_i(\mathbf{x}_i)} + |H_i|\right). \end{aligned}$$

Using Lemma 17 and the fact that $\sum_{i \in \mathcal{N}} |H_i| \leq n$, we conclude that $\frac{\mathsf{OPT}}{\mathsf{ALG}} \leq 5$. ◀

3.2.2 Constant number of agents

We can prove the following theorem for NSW with a constant number of agents. This theorem also works for weighted NSW. We note that a similar result was known previously [32]. See Remark 21 for a comparison.

▶ **Theorem 20.** *Given an instance of the fair allocation problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i \in \mathcal{N}})$, we can compute in randomized polynomial time an allocation $A_i \in \Pi_n([m])$ such that the value of* NSW *of the allocation is $(1 - 1/e - \epsilon)$*OPT *where* OPT *is the value of the optimal Nash social welfare.*

Proof Sketch. The proof is similar to proof of constantly many agents for the max-min problem (see 14) and we only sketch the idea here. First, we predict the value agent i receives in an optimal allocation, say OPT_i. For a particular agent, we can guess this to a $(1+\epsilon)$ multiplicative approximation using a grid search in polynomial time. Therefore, for constantly many agents, we can guess the valuation profile for an optimal NSW solution in polynomial time. Now, consider the corresponding optimal allocation, $(S_1^*, \ldots, S_n^*)$. From each S_i^*, we guess a subset $A_i \subseteq S_i^*$ such that for each $g \in S_i^* \setminus A_i$, $v_i(g \mid A_i) \leq \epsilon\mathsf{OPT}_i$. It is clear that we can have an A_i such that $|A_i| \leq (1/\epsilon)$ otherwise, we have a value of more than OPT_i for agent i. Therefore, together for all n agents, this A_i can be guessed in $m^{O(n/\epsilon)}$ time. This is clearly polynomial when n and ϵ are constants. Now, we find a fractional solution for the NSW problem – for each agent, we define it's new valuation function to be $v_{i|A_i}$ which is the function that gives marginal values of sets on A_i. It is well-known and easy to see that this function is also submodular. For this function, we aim to allocate agent i value $\mathsf{OPT}_i - v_i(A_i)$. Again, following the Theorem 11 from previous section, if such an allocation exists, we can compute a fractional allocation $\mathbf{x}$ in time polynomial in m, n and ϵ' such that $v_i(\mathbf{x}_i) \geq (1-1/e-\epsilon')(\mathsf{OPT}_i - v_i(A_i))$. Further, we can also ensure that no agent is allocated a good of value more than $\epsilon \cdot \mathsf{OPT}$ by making the value of any such good for the agent zero (this also maintains submodularity). Then we run the non-uniform pipage rounding algorithm and obtain an allocation that allocates each agent a value of $(1-1/e-\epsilon'-\epsilon)\mathsf{OPT}_i$. Therefore, in time polynomial in m, n and $\epsilon'' = \epsilon' + \epsilon$, we can compute an allocation that maximizes NSW with constant number of agent up to a factor of $(1-1/e-\epsilon'')$ i.e., we have a PTAS for this problem with constantly many agents. Note that this algorithm also works with asymmetric agents. ◀

▶ Remark 21. A $(1-1/e-\epsilon)$-approximation for constant n was achieved previously [32]. That work relied on an integral version of Theorem 11 which in turn was based on the same ideas as in max-min allocation from [20]. Therefore, to obtain an approximation guarantee of $(1-1/e-\epsilon)$ for any fixed $\epsilon > 0$, [32] requires guessing from an optimal allocation all goods allocated to an agent i of value at least $\epsilon \cdot \mathsf{OPT}_i/\log n$. In the preceding theorem, for a fixed $\epsilon > 0$, we need to guess the "large" goods of value at least $\epsilon\mathsf{OPT}_i$. Therefore the size of the set to be guessed is reduced.

3.3 Approximating MMS for Submodular Valuations

MMS is another fairness objective frequently studied in allocation problems. It is a share-based guarantee, i.e., each agent demands a certain value a.k.a. share. An allocation is considered to be fair if all agents receive their share. In case of MMS, this value for a particular agent i depends only on the valuation function of the agent, v_i and the number of agents $|\mathcal{N}|$. It is defined as follows:

$$\mathsf{MMS}_i^{|\mathcal{N}|}(\mathcal{M}) := \max_{A \in \Pi_{|\mathcal{N}|}(\mathcal{M})} \min_{i \in [n]} v_i(A_i).$$

Essentially, the MMS value of an agent is the maximum value she can ensure for herself when she divides the goods into $|\mathcal{N}|$ parts and chooses the worst part for herself. When $\mathcal{M}$ and $\mathcal{N}$ are clear from the context, we denote $\mathsf{MMS}_i^{|\mathcal{N}|}(\mathcal{M})$ as simply MMS_i. As mentioned in Section 1.1.3, since MMS-allocations do not exist, we study α-MMS allocations for $\alpha(\leq 1)$ as large as possible. Here, we give an algorithm that achieves a value of $\approx 1/2(1-1/e) \approx 0.316$ based on a simple application of the NonUniformPipageRounding. Our approach is again to round a fractional allocation. While in the previous two sections we solved some fractional

program to find the fractional allocation, in this section, we give a direct solution using two known results. To state the results, we need to define another extension of submodular valuation functions called the concave extension. Given a submodular function, f the concave extension, f^+ of f is defined as follows.

▶ **Definition 22** (Concave extension of f)**.**

$$f^+(\mathbf{x}) = \max_{(\alpha_S)_{S\in 2^{\mathcal{M}}}} \{\sum_S f(S)\alpha_S : \sum_S \alpha_S = 1 \text{ and } \sum_{S\ni j} \alpha_S = x_j \quad \forall j \in \mathcal{M} \text{ and } \alpha_S \geq 0 \quad \forall S \in 2^{\mathcal{M}}\}.$$

Now, we state the aforementioned results. The first one, from [19] gives an upper bound on MMS in terms of the concave extension. The second one, from [14], is called the correlation gap and connects the value of the concave extension at any fractional point with value of the multilinear extension at the same point.

▶ **Lemma 23** ([19])**.** *Given an instance of the fair division problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ with submodular valuations, $\mathsf{MMS}_i \leq v_i^+(\frac{1}{|\mathcal{N}|}(\mathbf{1}^{|\mathcal{M}|}))$ for all $i \in \mathcal{N}$, where v_i^+ is the concave extension of the function v_i.*

▷ Claim 24 ([14]). For any monotone submodular function, f with concave extension f^+ and multilinear extension F,

$$F(\mathbf{x}) \geq \left(1 - \frac{1}{e}\right) f^+(\mathbf{x}).$$

Combining the above two results, we get,

▷ Claim 25. Given an instance of the fair division problem, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ with submodular valuations, $\mathsf{MMS}_i \leq \left(\frac{e}{e-1}\right) \cdot V_i(\frac{1}{|\mathcal{N}|}(\mathbf{1}^{|\mathcal{M}|}))$.

We can now use NonUniformPipageRounding to obtain:

▶ **Theorem 26.** *Given a fair division instance, $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ with submodular valuations, if $\max_{i,j} v_i(\{j\}) \leq \epsilon \cdot \mathsf{MMS}_i$ for all $i \in \mathcal{N}, j \in \mathcal{M}$, we can compute an allocation in randomized polynomial time where each agent receives a value of $\left(1 - \frac{1}{e} - \epsilon\right) \mathsf{MMS}_i$.*

Proof. We run NonUniformPipageRounding on the allocation $\mathbf{x} = (\frac{1}{|\mathcal{N}|}\mathbf{1}^{|\mathcal{M}|})$. Therefore, we get an integral allocation, $\mathcal{A} = (A_1, \ldots, A_{|\mathcal{N}|})$ such that $v_i(A_i) \geq V_i(\frac{1}{|\mathcal{N}|}(\mathbf{1}^{|\mathcal{M}|})) - v_i(\ell_i)$ where $\ell_i \coloneqq \operatorname{argmax}_j v_i(\{j\})$. Since by assumption $v_i(\ell_i) \leq \epsilon\mathsf{MMS}_i$ by assumption, and from Claims 24 and 25, $V_i(\frac{1}{|\mathcal{N}|}(\mathbf{1}^{|\mathcal{M}|})) \geq \left(1 - \frac{1}{e}\right) \mathsf{MMS}_i$, the theorem follows. ◀

Algorithm 4 gives a ≈ 0.316-approximation for MMS with submodular valuations. To analyze this, we first understand the algorithm. Step (6) is performing our NonUniformPipageRounding. The while loop in lines (2) to (4) performs an operation that is called as "single good reduction". The following claim regarding this is well-known and used in almost all works on MMS.

▷ Claim 27. Given a fair division instance $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$, the MMS value of any agent is retained if we remove any single agent and any single good. That is,

$$\mathsf{MMS}_i^{|\mathcal{N}|}(\mathcal{M}) \leq \mathsf{MMS}_i^{|\mathcal{N}|-1}(\mathcal{M} \setminus \{g\}) \qquad \text{for all } g \in \mathcal{M}$$

We can now prove the following result for Algorithm 4.

Algorithm 4 $\frac{1}{2}(1-\frac{1}{e}-o(1))$-approximation for MMS with submodular valuations.

Input : Instance $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$
Output : An allocation where each agent receives at least $\frac{1}{2}(1-\frac{1}{e}-o(1))\mathsf{MMS}_i$.

1 Initialize $\mathcal{N}' \leftarrow \mathcal{N}$ and $\mathcal{M}' \leftarrow \mathcal{M}$
2 **while** *there exists* $i \in \mathcal{N}'$ *and* $g \in \mathcal{M}'$ *with* $v_i(g) \geq \frac{1}{2}V_i(\frac{1}{|\mathcal{N}'|}\mathbf{1}^{|\mathcal{M}'|})$ **do**
3 Allocate g to i
4 Update $\mathcal{N}' \leftarrow \mathcal{N}' \setminus i$ and $\mathcal{M}' \leftarrow \mathcal{M}' \setminus g$
5 Initialize $\mathbf{x} = \left(\frac{1}{|\mathcal{N}'|}(\mathbf{1}^{|\mathcal{N}'|\times|\mathcal{M}'|})\right)$
6 $(A_1, \ldots, A_{|\mathcal{N}'|}) \leftarrow$ NonUniformPipageRound($\mathbf{x}$)
7 **return** $(A_1, \ldots, A_{|\mathcal{N}|})$

▶ **Theorem 28.** *Given a fair division instance,* $(\mathcal{N}, \mathcal{M}, (v_i)_{i\in\mathcal{N}})$ *with submodular valuations, Algorithm 4 outputs an allocation where each agent receives a value of* $\frac{1}{2}\left(1-\frac{1}{e}-o(1)\right)\mathsf{MMS}_i$.

Proof. First, let us assume that we can compute $V_i(\mathbf{y})$ exactly for any $\mathbf{y} \in [0,1]^{|\mathcal{N}|\times|\mathcal{M}|}$. Note that as per Algorithm 4, an agent can receive an allocation either in the while loop of Steps 2-4 or in Step 6 via NonUniformPipageRounding. In every iteration of the while loop, we remove one agent and one good. Therefore, by Claim 27, the MMS value of any agent in the residual instance does not decrease. As a result, until the algorithm reaches Step 6, we always have $\mathsf{MMS}_i \leq (\frac{e}{e-1})V_i(\frac{1}{|\mathcal{N}'|}\mathbf{1}^{|\mathcal{M}'|})$. Therefore, any agent who gets a good in the while loop receives a good of value at least $\frac{1}{2}(1-\frac{1}{e})\mathsf{MMS}_i$. Now, in Step 6, by Lemma 8, we get an allocation of value at least $V_i(\frac{1}{|\mathcal{N}'|}\mathbf{1}^{|\mathcal{M}'|}) - v_i(\ell_i) \geq \frac{1}{2}V_i(\frac{1}{|\mathcal{N}'|}\mathbf{1}^{|\mathcal{M}'|}) \geq \frac{1}{2}(1-\frac{1}{e})\mathsf{MMS}_i$.

Now, we note that the values of $V_i(\frac{1}{|\mathcal{N}'|}\mathbf{1}^{|\mathcal{M}'|})$ are computed by sampling and in polynomial time, we can only compute them to a factor of $(1-\epsilon)$. Accounting for this, we get that in randomized polynomial time we can guarantee each agent a value of $(1-\frac{1}{e}-o(1))\mathsf{MMS}_i$. This proves the theorem. ◀

References

1 Hannaneh Akrami and Jugal Garg. Breaking the 3/4 barrier for approximate maximin share. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 74–91. SIAM, 2024. doi:10.1137/1.9781611977912.4.

2 Hannaneh Akrami, Jugal Garg, Eklavya Sharma, and Setareh Taki. Improving approximation guarantees for maximin share. In *25th ACM Conference on Economics and Computation*, pages 198–198. ACM, 2024.

3 Georgios Amanatidis, Evangelos Markakis, Afshin Nikzad, and Amin Saberi. Approximation algorithms for computing maximin share allocations. *ACM Transactions on Algorithms (TALG)*, 13(4):1–28, 2017. doi:10.1145/3147173.

4 Nima Anari, Shayan Oveis Gharan, Amin Saberi, and Mohit Singh. Nash social welfare, matrix permanent, and stable polynomials. *arXiv preprint arXiv:1609.07056*, 2016. arXiv:1609.07056.

5 Nima Anari, Tung Mai, Shayan Oveis Gharan, and Vijay V. Vazirani. Nash social welfare for indivisible items under separable, piecewise-linear concave utilities. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 2274–2290. SIAM, 2018. doi:10.1137/1.9781611975031.147.

6 Etienne Bamas, Paritosh Garg, and Lars Rohwedder. The Submodular Santa Claus Problem in the Restricted Assignment Case. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*, volume 198 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:18, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2021.22.

7 Étienne Bamas, Sarah Morell, and Lars Rohwedder. The submodular Santa Claus problem. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 616–640. SIAM, 2025. doi:10.1137/1.9781611978322.19.

8 Nikhil Bansal and Maxim Sviridenko. The santa claus problem. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 31–40, 2006. doi:10.1145/1132516.1132522.

9 Siddharth Barman and Sanath Kumar Krishnamurthy. Approximation algorithms for maximin fair division. *ACM Transactions on Economics and Computation (TEAC)*, 8(1):1–28, 2020. doi:10.1145/3381525.

10 Siddharth Barman, Sanath Kumar Krishnamurthy, and Rohit Vaish. Finding fair and efficient allocations. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 557–574, 2018. doi:10.1145/3219166.3219176.

11 Xiaohui Bei, Yuda Feng, Yang Hu, Shi Li, and Ruilong Zhang. Nash social welfare with submodular valuations: Approximation algorithms and integrality gaps. *arXiv preprint arXiv:2504.09669*, 2025. To appear in Proc. of ACM STOC 2026. doi:10.48550/arXiv.2504.09669.

12 Ivona Bezáková and Varsha Dani. Allocating indivisible goods. *ACM SIGecom Exchanges*, 5(3):11–18, 2005. doi:10.1145/1120680.1120683.

13 Eric Budish. The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*, 119(6):1061–1103, 2011.

14 Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 182–196. Springer, 2007.

15 Gruia Calinescu, Chandra Chekuri, Martin Pal, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011. doi:10.1137/080733991.

16 Ioannis Caragiannis, David Kurokawa, Hervé Moulin, Ariel D Procaccia, Nisarg Shah, and Junxing Wang. The unreasonable fairness of maximum nash welfare. *ACM Transactions on Economics and Computation (TEAC)*, 7(3):1–32, 2019. doi:10.1145/3355902.

17 Deeparnab Chakrabarty, Julia Chuzhoy, and Sanjeev Khanna. On allocating goods to maximize fairness. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 107–116. IEEE, 2009. doi:10.1109/FOCS.2009.51.

18 Chandra Chekuri, T.S. Jayram, and Jan Vondrak. On multiplicative weight updates for concave and submodular function maximization. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, ITCS '15, pages 201–210, New York, NY, USA, 2015. Association for Computing Machinery. doi:10.1145/2688073.2688086.

19 Chandra Chekuri, Pooja Kulkarni, Rucha Kulkarni, and Ruta Mehta. 1/2-approximate mms allocation for separable piecewise linear concave valuations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38(9), pages 9590–9597, 2024. doi:10.1609/AAAI.V38I9.28815.

20 Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 575–584. IEEE, 2010. doi:10.1109/FOCS.2010.60.

21 Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 783–792, 2011. doi:10.1145/1993636.1993740.

22 Siu-Wing Cheng and Yuchen Mao. Restricted Max-Min Allocation: Approximation and Integrality Gap. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.38.

23 Richard Cole, Nikhil Devanur, Vasilis Gkatzelis, Kamal Jain, Tung Mai, Vijay V Vazirani, and Sadra Yazdanbod. Convex program duality, fisher markets, and nash social welfare. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 459–460, 2017. doi:10.1145/3033274.3085109.

24 Richard Cole and Vasilis Gkatzelis. Approximating the nash social welfare with indivisible items. *SIAM Journal on Computing*, 47(3):1211–1236, 2018. doi:10.1137/15M1053682.

25 Shahar Dobzinski and Michael Schapira. An improved approximation algorithm for combinatorial auctions with submodular bidders. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 1064–1073, 2006. URL: http://dl.acm.org/citation.cfm?id=1109557.1109675.

26 Yuda Feng, Yang Hu, Shi Li, and Ruilong Zhang. Constant approximation for weighted nash social welfare with submodular valuations. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 1395–1405, 2025. doi:10.1145/3717823.3718203.

27 Yuda Feng and Shi Li. A note on approximating weighted nash social welfare with additive valuations. *TheoretiCS*, 4, 2025. doi:10.46298/THEORETICS.25.17.

28 Jugal Garg, Martin Hoefer, and Kurt Mehlhorn. Approximating the nash social welfare with budget-additive valuations. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2326–2340. SIAM, 2018. doi:10.1137/1.9781611975031.150.

29 Jugal Garg, Edin Husić, Wenzheng Li, László A Végh, and Jan Vondrák. Approximating nash social welfare by matching and local search. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1298–1310, 2023.

30 Jugal Garg, Edin Husić, and László A Végh. Approximating nash social welfare under rado valuations. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1412–1425, 2021.

31 Jugal Garg, Edin Husić, Wenzheng Li, László A. Végh, and Jan Vondrák. Approximating nash social welfare by matching and local search, 2026. arXiv:2211.03883.

32 Jugal Garg, Pooja Kulkarni, and Rucha Kulkarni. Approximating nash social welfare under submodular valuations through (un) matchings. *ACM Transactions on Algorithms*, 19(4):1–25, 2023. doi:10.1145/3613452.

33 Jugal Garg, Peter McGlaughlin, and Setareh Taki. Approximating Maximin Share Allocations. In *2nd Symposium on Simplicity in Algorithms (SOSA 2019)*, volume 69 of *Open Access Series in Informatics (OASIcs)*, pages 20:1–20:11. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/OASIcs.SOSA.2019.20.

34 Jugal Garg and Setareh Taki. An improved approximation algorithm for maximin shares. In *Proceedings of the 21st ACM Conference on Economics and Computation*, pages 379–380, 2020. doi:10.1145/3391403.3399526.

35 Mohammad Ghodsi, MohammadTaghi HajiAghayi, Masoud Seddighin, Saeed Seddighin, and Hadi Yami. Fair allocation of indivisible goods: Improvements and generalizations. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 539–556, 2018.

36 Mamoru Kaneko and Kenjiro Nakamura. The nash social welfare function. *Econometrica: Journal of the Econometric Society*, pages 423–435, 1979.

37 Euiwoong Lee. Apx-hardness of maximizing nash social welfare with indivisible items. *Inf. Process. Lett.*, 122:17–20, 2017. doi:10.1016/J.IPL.2017.01.012.

38 Benny Lehmann, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. In *Proceedings of the 3rd ACM conference on Electronic Commerce*, pages 18–28, 2001. doi:10.1145/501158.501161.

39 Jan Karel Lenstra, David B Shmoys, and Éva Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical programming*, 46:259–271, 1990. doi:10.1007/BF01585745.

40 Wenzheng Li and Jan Vondrák. A constant-factor approximation algorithm for nash social welfare with submodular valuations. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 25–36. IEEE, 2022.

41 Li Lijun, Xu Chenyang, Yang Liuyi, and Ruilong Zhang. Fair submodular maximization over a knapsack constraint. In *34th International Joint Conference on Artificial Intelligence*, 2025.

42 Richard J Lipton, Evangelos Markakis, Elchanan Mossel, and Amin Saberi. On approximately fair allocations of indivisible goods. In *Proceedings of the 5th ACM Conference on Electronic Commerce*, pages 125–131, 2004. doi:10.1145/988772.988792.

43 Vahab Mirrokni, Michael Schapira, and Jan Vondrák. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In *Proceedings of the 9th ACM conference on Electronic commerce*, pages 70–77, 2008. doi:10.1145/1386790.1386805.

44 George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978. doi:10.1007/BF01588971.

45 Serge A Plotkin, David B Shmoys, and Éva Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20(2):257–301, 1995. doi:10.1287/MOOR.20.2.257.

46 Ariel D Procaccia and Junxing Wang. Fair enough: Guaranteeing approximate maximin shares. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 675–692, 2014. doi:10.1145/2600057.2602835.

47 David B Shmoys and Éva Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical programming*, 62(1):461–474, 1993. doi:10.1007/BF01585178.

48 Gilad Ben Uziahu and Uriel Feige. On fair allocation of indivisible goods to submodular agents. *arXiv preprint arXiv:2303.12444*, 2023. doi:10.48550/arXiv.2303.12444.

49 Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 67–74, 2008. doi:10.1145/1374376.1374389.

A Configuration-LP Framework for Connected k-Median Clustering

Kushagra Chatterjee[1]
Indian Statistical Institute, Kolkata, India

Rojin Rezvan
Virginia Tech, Blacksburg, VA, USA

Ali Vakilian
Virginia Tech, Blacksburg, VA, USA

Abstract

We study the *connected k-median* clustering problem, a clustering problem that augments the classical k-median objective with connectivity constraints. We focus on the *overlapping* variant of the problem, where clusters are allowed to share vertices. In addition to a metric space (V, d), the input contains a connected graph G on the same vertex set V of size n. The goal is to select at most k centers C and assign vertices to them so as to minimize the k-median cost (i.e., $\sum_{v \in V} d(v, C)$), subject to the constraint that each cluster induces a connected subgraph of G. Since the metric space and the connectivity graph are independent, the problem is significantly more challenging than standard clustering. Eube et al. [29] showed that even the assignment version is $\Omega(\log n)$-hard to approximate and gave approximation algorithms with guarantees depending polynomially on k.

We develop a configuration-LP-based framework that combines covering LP techniques with a rooted minimum-density oracle. For the assignment version, we obtain an $O(\log^2 n)$-approximation. For the general version, we develop a bicriteria framework that opens $O(k \log n)$ centers while achieving an $O(\log^2 n)$-approximation in cost.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Theory of computation → Facility location and clustering

Keywords and phrases Connected Clustering, k-Median Clustering, Configuration LP

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.22

Category APPROX

Funding *Ali Vakilian*: Supported in part by NSF CAREER Award CCF-2542925.

1 Introduction

Clustering is a central problem in algorithms and data analysis, with classical objectives such as k-center [35, 39] and k-median [15, 14, 19] extensively studied over the past decades. In the k-median problem, we are given a vertex set V of size n together with a metric $d : V \times V \to \mathbb{R}_{\geq 0}$ and an integer k, and the goal is to select k centers so as to minimize the sum of distances of each vertex to its assigned center. While the vanilla k-median problem admits constant-factor approximation algorithms [15, 14, 5, 23], many modern applications require incorporating additional structural constraints into the clustering process.

In this work, we study the *connected k-median* problem, an extension of the classical k-median problem that incorporates *connectivity constraints*. In addition to the metric space (V, d), the input includes an unweighted undirected graph $G = (V, E)$, referred to as the *connectivity graph*. The goal is to select k centers and assign vertices to them so as

[1] This work was carried out while the author was a PhD student at the National University of Singapore.


© Kushagra Chatterjee, Rojin Rezvan, and Ali Vakilian;
licensed under Creative Commons License CC-BY 4.0



to minimize the k-median objective, subject to the constraint that each cluster induces a connected subgraph in G. Importantly, the graph G is completely independent of the metric d, making the problem fundamentally different from classical clustering.

We focus on the *overlapping* variant of the connected k-median problem, where a vertex may belong to more than one cluster. This is natural in applications such as community detection [8, 30], where individuals can participate in multiple groups, and it also allows vertices such as articulation points to serve as connectors between several clusters. Theoretically, this distinction is crucial; Eube et al. [29] showed that the disjoint version is extremely hard to approximate, admitting an $\Omega(n^{1-\varepsilon})$ lower bound even for $k = 2$.

Connected clustering arises naturally in settings where both *attribute data* and *relational structure* are present. For instance, in applications such as community detection in social networks [8, 30] or spatial partitioning tasks like school redistricting [18], the input consists of data points equipped with features as well as an underlying graph capturing relationships such as social interactions or geographic adjacency. In such settings, it is desirable to form clusters that are not only metrically coherent but also respect the connectivity structure of the graph. For example, in school redistricting, geographical units must be grouped into contiguous regions, ensuring that each cluster forms a connected area, while still optimizing criteria such as proximity or compactness. This motivates the study of clustering models that integrate geometric objectives with graph-theoretic constraints.

From an algorithmic perspective, connectivity constraints significantly increase the difficulty of clustering problems. Even in the *assignment version* of connected k-median, where the set of centers is fixed, and the goal is only to assign vertices to centers, Eube et al. [29] showed that the problem is $\Omega(\log n)$-hard to approximate, and provided an $O(k \log n)$-approximation algorithm. For the general version, where centers must also be selected, they obtained an $O(k^2 \log n)$-approximation. These results demonstrate a substantial gap between k-median clustering and its connected variant.

Our contributions. Our main contribution is a configuration-LP framework for connected k-median clustering. Unlike the cut-based relaxation of Eube et al. [29], where connectivity is enforced indirectly through constraints, our LP uses variables indexed by connected sets rooted at centers, thereby encoding connectivity directly into the variables themselves. We combine this formulation with covering-LP techniques and rooted minimum-density oracles to obtain approximation guarantees for connected k-median. Our main results are summarized below.

- **Assignment version.** We obtain an $O(\log^2 n)$-approximation algorithm for general graphs, which improves Eube et al [29] assignment version result when $k = \omega(\log n)$.
- **General version (bicriteria approximation).** We develop a bicriteria framework that allows opening more than k centers. In particular, we compute a solution that opens at most $O(k \log n)$ centers while achieving an $O(\log^2 n)$-approximation in cost, yielding an $(O(\log n), O(\log^2 n))$ bicriteria approximation.

The bicriteria guarantee highlights a tradeoff between the number of opened centers and the approximation factor in cost. The algorithm of Eube et al. [29] gives a solution with k centers with approximation factor of $O(k^2 \log n)$ for the general version. In contrast, our bicriteria framework opens $O(k \log n)$ centers and achieves an $O(\log^2 n)$ cost approximation, eliminating the polynomial dependence on k in the objective guarantee.

1.1 Technical Overview

Our algorithms are based on a configuration-LP viewpoint. Instead of assigning vertices to centers directly, we introduce a variable for each pair (c, T), where c is a center and T is a connected vertex set containing c. Choosing such a configuration corresponds to opening a connected cluster rooted at c. This formulation captures the connectivity requirement directly: every integral solution selects connected sets, and the covering constraints ensure that every vertex is contained in at least one selected cluster. It is useful to contrast this formulation with the cut-based LP used by Eube et al. [29]. In the cut-LP, the variables x_v^c represent fractional assignments of vertices to centers, and connectivity is enforced indirectly through cut constraints: if v is assigned to c, then every v–c cut must contain enough mass assigned to c. This is a natural and compact relaxation, but it is still a *vertex-assignment* formulation; after rounding, one must still ensure that the assigned vertices form connected clusters. In contrast, the configuration LP works directly with connected sets. A variable $y_{T,c}$ corresponds to selecting an entire connected cluster T rooted at c, so connectivity is built into the variables themselves rather than enforced only through constraints.

This global representation is particularly useful for rounding. Once a configuration (c, T) is sampled, it already defines a connected set containing its center, and the union of sampled configurations with the same root remains connected. Thus, the randomized rounding step only needs to ensure coverage of vertices; connectivity is preserved automatically. Moreover, every configuration-LP solution induces a feasible solution to the cut-LP by setting $x_v^c = \sum_{\substack{T \in \mathcal{T}_c \\ v \in T}} y_{T,c}$, with the same objective value, so the configuration LP can be viewed as a more structured relaxation. The price of this stronger representation is that the LP contains exponentially many variables, so solving it exactly is not straightforwardly achievable in polynomial time. In fact, we can only solve this LP relaxation approximately, using column generation based on the rooted minimum-density connected-set oracle.

For the assignment version, where the center set C is fixed, the configuration LP is a pure covering LP. We solve it approximately using the multiplicative weight update framework of Plotkin, Shmoys and Tarods [43] for covering linear programs. The key step is the pricing problem: given weights α_v on the vertices that correspond to the covering constraints, we need to find a connected set $T \ni c$ minimizing the ratio between its cost and the total weight of vertices it covers. This is exactly a rooted minimum-density connected-set problem:

$$\min_{T | T \ni c} \frac{\sum_{v \in T} d(v, c)}{\sum_{v \in T} \alpha_v}.$$

Thus, a ρ-approximate oracle for rooted minimum-density connected sets yields a ρ-approximate pricing oracle for the configuration LP. Combining this with the Plotkin–Shmoys–Tardos framework gives a polynomial-support fractional solution whose cost is within a factor $O(\rho)$ of the optimum LP fractional value. We obtain such density oracles using connections to the *node-weighted Steiner tree* problem.

The rounding step is simple and uses the standard independent randomized rounding. Given a fractional solution $\hat{y}$, we sample each configuration (c, T) with probability proportional to $\hat{y}_{T,c} \log n$. For each center c, the final cluster is the union of all the sampled configurations rooted at c. This union remains connected because every sampled set contains the common root c. The logarithmic oversampling ensures, by a union bound over all vertices, that every vertex is covered with high probability. Linearity of expectation bounds the expected cost by an additional $O(\log n)$ factor. This yields an $O(\rho \log n)$-approximation for the assignment version. For general graphs, the rooted minimum-density problem admits an $O(\log n)$-approximation via known algorithms for the *budgeted node-weighted Steiner tree* problem.

This yields an $O(\log^2 n)$-approximation for the assignment version. For minor-closed graphs, the techniques of [25, 17] suggest that the rooted minimum-density problem may admit an $O(1)$-approximation, which would improve the assignment-version guarantee to $O(\log n)$.

For the general version, we provide a bicriteria approximation guarantee. We avoid the center-rounding step of Eube et al [29] and instead formulate a non-fixed-center configuration LP. This LP has variables for all roots $c \in V$ and all connected sets $T \in \mathcal{T}_c$, together with a fractional center-budget constraint $\sum_{c \in V} \sum_{T \in \mathcal{T}_c} y_{T,c} \leq k$. This LP is a mixed packing/covering program. Its pricing problem again reduces to a rooted minimum-density connected-set problem, except that the ratio includes both the connection cost of the configuration and the multiplier corresponding to the center-budget constraint. After computing a near-optimal fractional solution, we apply the same randomized rounding scheme. The covering constraints guarantee that all vertices are covered with high probability, while the budget constraint implies that the number of sampled configurations, and hence the number of opened centers, is at most $O(k \log n)$ with high probability. This gives an $(O(\log n), O(\rho \log n))$ bicriteria approximation, which becomes $(O(\log n), O(\log^2 n))$ for general graphs.

1.2 Related Work

Clustering problems such as k-center and k-median are among the most well-studied optimization problems in algorithm design. Both problems are known to be NP-hard. For the k-center problem, a tight 2-approximation algorithm is known [33, 35], and this factor is optimal unless P=NP [36]. For metric k-median, the current best polynomial-time approximation guarantee is $2 + \varepsilon$, for any fixed $\varepsilon > 0$, achieved by the recent algorithm of Cohen-Addad et al. [22]. In contrast, no polynomial-time algorithm can achieve an approximation ratio smaller than $1 + 2/e \approx 1.736$, unless P = NP [37].

Connected clustering. Clustering with connectivity constraints has been studied in several contexts. Ge et al. [31] introduced the connected k-center problem as a model for jointly clustering attribute data and relationship data: the metric captures similarity between attribute vectors, while the connectivity graph captures relational information. They gave an exact dynamic program when the connectivity graph is a tree, and also studied general graphs, although without a known approximation guarantee. For general graphs with disjoint clusters, Drexler et al. [26] obtained an $O(\log^2 k)$-approximation; they also gave constant-factor approximations for Euclidean metrics of constant dimension and metrics of constant doubling dimension, motivated by applications in geodesy and sea-level analysis. In contrast, when overlapping clusters are allowed, connected k-center admits a simple 2-approximation, highlighting an important distinction between the disjoint and overlapping models.

For the k-median objective, the connected variant has been studied more recently. Validi et al. [46] consider connected k-median in the context of districting, where geographical units must be grouped into contiguous regions while optimizing a proximity-based objective. They formulate the problem as an integer linear program, which can be solved optimally but requires exponential time in the worst case. Gupta et al. [34] study the inapproximability of clustering with connectivity constraints and show that even when $k = 2$ and the connectivity graph is star-like, the problem is $\Omega(\log n)$-hard to approximate. They also provide an $O(\log n)$-approximation algorithm for some restricted instances of the problem.

Clustering with additional constraints. Beyond connectivity, a large body of work has focused on incorporating additional constraints into clustering problems motivated by real-world applications. Capacitated clustering introduces upper bounds on cluster sizes [40], while

models with outliers allow for excluding a small fraction of points from the clustering [16]. More recently, fairness constraints have been extensively studied, requiring clusters to satisfy various fairness conditions [21, 9, 6, 10, 24, 38, 41, 45, 32, 1, 42, 28, 27, 20]. Fairness constraints have also been studied for other types of clustering problems, like correlation clustering [3, 2, 4] and consensus clustering [13, 11, 12]. These variants highlight the increasing importance of structured clustering models, where combinatorial constraints are integrated with geometric objectives.

2 Preliminaries and Problem Statement

In this section, we introduce the definitions and notation used throughout the paper. We also formally state the problem and present our main results.

Problem Statement. We are given a vertex set V, a metric $d : V \times V \to \mathbb{R}_{\geq 0}$, and a connected *connectivity graph* $G = (V, E)$, which is unrelated to d. We consider the problem in two settings.

- **Assignment Version:** A set of centers $C \subseteq V$ with $|C| \leq k$, for some integer k is given. A feasible *overlapping* clustering solution outputs connected sets $\mathcal{S} := (S_c)_{c \in C}$ such that (i) $c \in S_c$ for all $c \in C$, (ii) $G[S_c]$ is connected for all $c \in C$, and (iii) $\bigcup_{c \in C} S_c = V$.
- **General Version:** An integer k is given, and the goal is to choose a set of centers $C \subseteq V$ with $|C| \leq k$ together with connected sets $\mathcal{S} := (S_c)_{c \in C}$ satisfying the same conditions as above.

In both cases, the objective is to find a feasible solution $\mathcal{S}$ that minimizes

$$\text{cost}(\mathcal{S}) := \sum_{c \in C} \sum_{v \in S_c} d(v, c),$$

where a vertex assigned to multiple clusters contributes once per cluster. Here $G[S_c]$ denotes the subgraph of G induced by the vertex set S_c.

▶ Remark 1. Let $G_1, \ldots, G_\ell$ be the connected components of G. In the assignment version, if some component contains at least one vertex but no center, then the instance is infeasible. Otherwise, each component can be solved independently, and the solutions can be combined. Hence, without loss of generality, we assume throughout that G is connected and every vertex can reach some center.

▶ Remark 2 (Bounded aspect ratio). Throughout the paper we assume the metric d has polynomially bounded aspect ratio: if $D_{\max} := \max_{u,v \in V} d(u, v)$ and $\delta_{\min} := \min_{u \neq v \in V} d(u, v)$, we assume

$$\Delta := \frac{D_{\max}}{\delta_{\min}} = \text{poly}(n).$$

This is the standard bounded-spread condition used throughout the metric clustering and embedding literature. All stated polynomial running times in this paper are with respect to n, k, and $1/\varepsilon$ under this assumption.

Our Results. We now formally state our results. A feasible overlapping clustering solution for either version is referred to as a *connected k-median clustering* of the vertex set V. We say that a clustering $\mathcal{S}$ is an α-approximate connected k-median clustering if $\text{cost}(\mathcal{S}) \leq \alpha \cdot \text{cost}(\mathcal{S}^*)$, where $\mathcal{S}^*$ denotes an optimal connected k-median clustering.

For the assignment version, we obtain an improved approximation guarantee when $k = \omega(\log n)$. In particular, we show the following.

▶ **Theorem 3.** *Given a vertex set V, a metric $d : V \times V \to \mathbb{R}_{\geq 0}$, an undirected, unweighted connectivity graph $G = (V, E)$, and a set of centers $C \subseteq V$ with $|C| \leq k$, there exists a polynomial-time algorithm that computes an $O(\log^2 n)$-approximate connected k-median clustering.*

We now consider the general version of the problem, where the set of centers is not given and must be chosen as part of the solution. For the general version, we have obtained bicriteria approximation guarantees. For this, we need to define a bicriteria notion of approximation that allows the algorithm to open more than k centers. A clustering $\mathcal{S}$ is said to be a (β, α)-approximate connected k-median clustering if it uses at most βk centers and satisfies $\text{cost}(\mathcal{S}) \leq \alpha \cdot \text{cost}(\mathcal{S}^*)$, where $\mathcal{S}^*$ is an optimal solution that uses at most k centers.

We obtain the following bicriteria guarantees.

▶ **Theorem 4.** *Given a vertex set V, a metric $d : V \times V \to \mathbb{R}_{\geq 0}$, an undirected, unweighted connectivity graph $G = (V, E)$ and an integer k, there exists a polynomial-time algorithm that computes an $(O(\log n), O(\log^2 n))$-approximate connected k-median clustering.*

3 Assignment Version

In this section, we prove our results for the assignment version of the problem. In particular, we prove Theorem 3, which we restate below.

▶ **Theorem 3.** *Given a vertex set V, a metric $d : V \times V \to \mathbb{R}_{\geq 0}$, an undirected, unweighted connectivity graph $G = (V, E)$, and a set of centers $C \subseteq V$ with $|C| \leq k$, there exists a polynomial-time algorithm that computes an $O(\log^2 n)$-approximate connected k-median clustering.*

To prove Theorem 3, we employ a *configuration LP* formulation. Since this LP has exponentially many variables, we compute a near-optimal fractional solution using a *rooted minimum-density connected set oracle* in conjunction with the Plotkin–Shmoys–Tardos [43] multiplicative weight update framework for covering LPs. We then round this fractional solution to obtain the desired approximation guarantee. The details are presented in the subsequent sections.

3.1 Configuration LP

We follow a *configuration LP* route: formulate an exponential-size covering LP over all connected clusters, solve it approximately via column generation/packing-covering with a *minimum-density* oracle, and round it in one shot via randomized rounding. This yields an $O(\rho \log n)$ approximation given a ρ-approximation oracle for the rooted minimum-density connected set problem.

For each center $c \in C$, let $\mathcal{T}_c$ denote the family of all connected vertex sets $T \subseteq V$ such that $c \in T$ and $G[T]$ is connected. For $T \in \mathcal{T}_c$, define

$$\text{cost}_c(T) := \sum_{v \in T} d(v, c).$$

We introduce a variable $y_{T,c} \geq 0$ for each pair (c, T) to represent the extent to which we choose configuration (c, T). We define the primal and dual relaxation of the assignment problem as follows.

Primal (Configuration) LP

$$\min \quad \sum_{c\in C}\sum_{T\in\mathcal{T}_c} \mathrm{cost}_c(T)\, y_{T,c} \tag{1}$$

$$\text{s.t.} \quad \sum_{c\in C}\sum_{\substack{T\in\mathcal{T}_c\\ v\in T}} y_{T,c} \geq 1 \qquad \forall v \in V \setminus C, \tag{2}$$

$$y_{T,c} \geq 0 \qquad \forall c \in C,\ \forall T \in \mathcal{T}_c. \tag{3}$$

Dual LP

$$\max \quad \sum_{v\in V\setminus C} \alpha_v \tag{4}$$

$$\text{s.t.} \quad \sum_{v\in T\setminus\{c\}} \alpha_v \leq \mathrm{cost}_c(T) \qquad \forall c \in C,\ \forall T \in \mathcal{T}_c, \tag{5}$$

$$\alpha_v \geq 0 \qquad \forall v \in V \setminus C. \tag{6}$$

▶ **Lemma 5.** *The configuration LP* (1)–(3) *is a valid relaxation of the assignment version of connected k-median with overlapping clusters. In particular, its optimal value is at most* OPT*.*

Proof. We provide the proof in the appendix. ◀

3.2 The Oracle: Rooted Minimum-Density Connected Set

Since the configuration LP has exponentially many variables (one for each connected set $T \in \mathcal{T}_c$ and each center $c \in C$), it is not immediate how to solve it in polynomial time. We follow the standard oracle-based approach for large-scale packing/covering linear programs: rather than enumerating all configurations explicitly, we access them through an optimization subroutine that, for suitable dual weights, returns an approximately best column [43, 47]. In our setting, this subroutine is the *rooted minimum-density connected-set problem* defined below.

▶ **Definition 6.** *Given a graph $G = (V, E)$, a root $c \in V$, nonnegative node* profits *$\alpha : V \to \mathbb{R}_{\geq 0}$, and nonnegative node* costs *$w : V \to \mathbb{R}_{\geq 0}$, the rooted minimum-density problem asks for a connected set $T \ni c$ minimizing*

$$\mathrm{dens}(T) := \frac{\sum_{v\in T} w(v)}{\sum_{v\in T} \alpha_v},$$

with the convention $\mathrm{dens}(T) = +\infty$ *if* $\sum_{v\in T} \alpha_v = 0$.

In our application, the dual variables are defined only on $V \setminus C$, so we set $\alpha_c := 0$. Thus $\sum_{v\in T} \alpha_v = \sum_{v\in T\setminus\{c\}} \alpha_v$, which matches the left-hand side of the dual packing constraint (5). In particular, minimizing $\mathrm{dens}(T)$ identifies the most violated dual constraint, and hence the most promising column for the primal configuration LP.

▶ **Definition 7.** *A ρ-approximate density oracle returns a connected set $T \ni c$ such that* $\mathrm{dens}(T) \leq \rho \cdot \min_{T'\ni c} \mathrm{dens}(T')$.

▶ **Lemma 8.** *There is a polynomial-time $O(\log n)$-approximate density oracle.*

Proof. The proof uses a result by Bateni et al [7]. We provide the full proof in the appendix. ◀

3.3 Computing a ρ-Approximate Solution of the Assignment Version of the Configuration LP

We next explain how to approximately solve the configuration LP using the rooted minimum-density oracle from the previous subsection. We follow the standard oracle-based approach for large covering LPs: we view the configuration LP as a covering problem over an implicit column set and apply the framework of Plotkin–Shmoys–Tardos [43]; the key point is that the corresponding pricing subproblem is exactly a rooted minimum-density connected-set problem. Therefore, a ρ-approximate density oracle yields, in polynomial time, a $(1+\varepsilon)\rho$-approximate fractional solution to the configuration LP.

▶ **Theorem 9.** *Assume a ρ-approximate density oracle. Then for any $\varepsilon \in (0,1)$, one can compute in polynomial time a feasible fractional solution $\hat{y}$ to the Configuration LP (1)–(3) satisfying*

$$\sum_{c\in C}\sum_{T\in\mathcal{T}_c} \mathrm{cost}_c(T)\hat{y}_{T,c} \le (1+\varepsilon)\rho\cdot \mathrm{OPT}_{\mathrm{CLP}} \le (1+\varepsilon)\rho\cdot \mathrm{OPT},$$

with support size $\mathrm{poly}(n, \varepsilon^{-1}, \rho)$.

Proof. We cast the configuration LP as a pure covering LP and apply a standard fractional packing/covering approximation scheme (Plotkin–Shmoys–Tardos [43]), whose guarantee is that an approximately optimal *fractional* solution can be computed given oracle access to an approximate *pricing* subproblem.

We discard all columns (c,T) with $T \cap (V \setminus C) = \emptyset$. Such columns cover no covering constraint and therefore play no role in the covering LP. After this restriction, every remaining column satisfies $\mathrm{cost}_c(T) > 0$.

Step 1: Configuration LP as a covering LP. Let $U_0 := V \setminus C$ and $n_0 := |U_0|$. Index columns by pairs (c,T) with $c \in C$ and $T \in \mathcal{T}_c$. Define the incidence matrix $A \in \{0,1\}^{n_0 \times N}$ and cost vector $c \in \mathbb{R}^N_{>0}$ by

$$A_{v,j} := \mathbf{1}[v \in T], \qquad c_j := \mathrm{cost}_c(T) = \sum_{u\in T} d(u,c).$$

Since d is a metric with $d(u,c) > 0$ for all $u \neq c$, every column $j = (c,T)$ with $T \neq \{c\}$ satisfies $c_j > 0$, so there are no $0/0$ anomalies in the pricing step. The configuration LP (1)–(3) is then $\min\{c^\top y : Ay \ge \mathbf{1}, y \ge 0\}$, a non-negative covering LP with $m := n_0$ constraints and N columns.

Step 2: Pricing is (approximate) rooted minimum-density. In the Plotkin–Shmoys–Tardos framework for covering LPs, the algorithm maintains a non-negative weight vector $p \in \mathbb{R}^{U_0}_{\ge 0}$ on constraints and, in each iteration, needs a column whose *cost per covered weight* is minimum (equivalently, whose covered weight per unit cost is maximum). For a column $j = (c,T)$, the covered weight under p is $\langle p, A_{\cdot,j}\rangle = \sum_{v\in T\cap U_0} p_v$, so the ratio is

$$\frac{c_j}{\langle p, A_{\cdot,j}\rangle} = \frac{\sum_{u\in T} d(u,c)}{\sum_{v\in T\cap U_0} p_v}.$$

This is the rooted minimum-density objective (Definition 6) with root c, costs $w_c(u) := d(u,c)$, and profits $\alpha_v := p_v$ (with $\alpha_c := 0$ for centers). A ρ-approximate density oracle therefore provides a ρ-approximate pricing oracle for the covering LP: calling it for each $c \in C$ and returning the column of globally smallest density gives an ρ-approximate minimizer of the ratio over all columns.

Step 3: Width and the upper bound q on $\mathrm{OPT}_{\mathrm{CLP}}$. To apply Theorem 10, we need (i) an upper bound $q \geq \mathrm{OPT}_{\mathrm{CLP}}$ and (ii) the associated width

$$W \geq q \cdot \max_{v \in U_0, j=(c,T)} \frac{A_{v,j}}{c_j} = q \cdot \max_{j=(c,T):T \neq \{c\}} \frac{1}{\mathrm{cost}_c(T)}. \tag{7}$$

Upper bound q. Construct the following explicit feasible solution: for each $v \in U_0$, let P_v be a shortest G-path from the nearest center c_v to v, which exists by the assumption on connectivity of G, and set $y_{V(P_v),c_v} = 1$ (with remaining variables 0). This is feasible (every v is covered) and has cost

$$q \coloneqq \sum_{v \in U_0} \mathrm{cost}_{c_v}(V(P_v)) \leq \sum_{v \in U_0} \sum_{u \in V(P_v)} d(u, c_v) \leq n_0 \cdot n \cdot D_{\max},$$

where $D_{\max} \coloneqq \max_{u,v \in V} d(u,v)$. Since the "Width W" step below normalizesd so that $\delta_{min} = 1$, then $D_{\max}$ conincides with the aspect ratio Δ, which is $\mathrm{poly}(n)$ by Remark 2. Therefore, $q = \mathrm{poly}(n)$.

Width W. After normalizing d by its minimum positive value so that $\delta \coloneqq \min_{u \neq v} d(u,v) = 1$ (this leaves all approximation ratios unchanged and ensures $c_j \geq 1$ for all non-trivial columns), the width in (7) becomes

$$W = q \cdot \max_j \frac{1}{c_j} \leq q \cdot \frac{1}{\delta} = q = \mathrm{poly}(n).$$

Step 4: Apply Theorem 10. With $m = n_0$, the ρ-approximate pricing oracle from Step 2, the upper bound $q \geq \mathrm{OPT}_{\mathrm{CLP}}$ from Step 3, and the width $W \leq q$ from Step 3, Theorem 10 yields a feasible $\hat{y}$ with

$$c^\top \hat{y} \leq \rho(1 + \varepsilon + \varepsilon^2)\mathrm{OPT}_{\mathrm{CLP}} \leq (1+\varepsilon)\rho \cdot \mathrm{OPT}_{\mathrm{CLP}}$$

absorbing ε^2 into ε by replacing ε with $\varepsilon/2$, and support size

$$|\mathrm{supp}(\hat{y})| = \widetilde{O}\left(\frac{n_0 W \rho}{\varepsilon^3}\right) = \widetilde{O}\left(\frac{n_0 q \rho}{\varepsilon^3}\right) = \mathrm{poly}(n).$$

Since $\mathrm{OPT}_{\mathrm{CLP}} \leq \mathrm{OPT}$ (Lemma 5), the stated bound $c^\top \hat{y} \leq (1+\varepsilon)\rho \cdot \mathrm{OPT}$ follows.

Runtime. Each iteration of the Plotkin–Shmoys–Tardos framework calls the density oracle $k \coloneqq |C|$ times (once per center) and selects the column of globally smallest density. The total number of iterations is $\widetilde{O}(n_0 W \rho/\varepsilon^3) = \mathrm{poly}(n)$. After all iterations, the restricted LP on the generated columns is solved exactly in polynomial time. Each oracle call is polynomial by Lemma 8, so the entire algorithm runs in polynomial time. ◀

The following theorem is a standard oracle-based reformulation of the fractional covering framework of Plotkin–Shmoys–Tardos; we cite Sharma [44, Theorems 3 and 4] for a clean restatement with approximate pricing and support bounds. For completeness, we provide a self-contained proof of Theorem 10 in the appendix.

▶ **Theorem 10** (Plotkin–Shmoys–Tardos [43]; cf. Sharma [44, Theorems 3 and 4])**.** *Consider the covering linear program* $\min\{c^\top y : Ay \geq \mathbf{1},\ y \geq 0\}$, *where* $A \in \mathbb{R}^{m \times N}_{\geq 0}$ *and* $c \in \mathbb{R}^N_{>0}$. *Let* OPT *denote its optimal value, and let* $q \geq \mathrm{OPT}$ *be a given upper bound on the optimal objective. Assume that for every weight vector* $w \in \mathbb{R}^m_{\geq 0}$, *there is a polynomial-time* α*-approximate pricing oracle returning a column* $j \in [N]$ *with*

$$\frac{c_j}{\sum_i A_{ij} w_i} \leq \alpha \cdot \min_{j' \in [N]} \frac{c_{j'}}{\sum_i A_{ij'} w_i}.$$

Let $W \geq q \max_{i,j}(A_{ij}/c_j)$ be a width bound. Then for every $\varepsilon \in (0,1]$, there is a polynomial-time algorithm that computes a feasible $\hat{y}$ with $c^\top \hat{y} \leq \alpha(1+\varepsilon+\varepsilon^2)\mathrm{OPT}$. Moreover, $|\mathrm{supp}(\hat{y})| = \widetilde{O}(mW\alpha/\varepsilon^3)$.

▶ Remark 11. The approximation ratio $\alpha(1+\varepsilon+\varepsilon^2)$ in Theorem 10 depends only on the oracle quality α and the accuracy parameter ε; it is *independent* of the width W. The width W affects only the support size and iteration count, not the approximation guarantee. In our setting $W \leq q = \mathrm{poly}(n)$, so both quantities are polynomial.

3.4 Rounding the Configuration LP

▶ **Theorem 12.** *Let $\hat{y}$ be any feasible solution to (1)–(3) with polynomial support. There is a randomized polynomial-time algorithm that, with high probability, outputs a feasible integral overlap solution $(S_c)c \in C$ covering all of $V \setminus C$ and its cost is $O(\log n) \cdot \sum_{c\in C}\sum_{T\in\mathcal{T}_c} \mathrm{cost}_c(T)\hat{y}_{T,c}$.*

Proof. Fix $\lambda \geq 2$. Independently for each pair (c,T) *in the support of* $\hat{y}$ (which is of polynomial size), sample T with probability $p_{T,c} \coloneqq \min\{1,\ \lambda \ln(n) \cdot \hat{y}_{T,c}\}$. For each $c \in C$, let $\mathcal{F}_c$ be the (multi)set of the sampled configurations and define $S_c \coloneqq \{c\} \cup \bigcup_{T\in\mathcal{F}_c} T$. Adding $\{c\}$ ensures $c \in S_c$ even if $\mathcal{F}_c = \emptyset$; since $d(c,c) = 0$, this adds zero cost. Each sampled T is connected and contains c, so S_c is a union of connected sets sharing c, hence connected.

Coverage. Fix $v \in V \setminus C$. Feasibility of $\hat{y}$ gives $\sum_{c,T:v\in T} \hat{y}_{T,c} \geq 1$. If any $p_{T,c} = 1$ for a pair with $v \in T$, then v is certainly covered. Otherwise $p_{T,c} = \lambda \ln(n)\hat{y}_{T,c}$ for all covering pairs, and

$$\Pr[v \text{ uncovered}] = \prod_{c,T:\, v\in T} (1 - p_{T,c}) \leq \exp\Big(-\sum_{c,T:\, v\in T} p_{T,c}\Big) \leq \exp(-\lambda \ln n_0) = n_0^{-\lambda}.$$

Union bound over $v \in V \setminus C$ (there are n_0 such vertices):

$$\Pr[\text{any } v \text{ uncovered}] \leq n_0 \cdot n_0^{-\lambda} \leq n_0^{-(\lambda-1)} \leq 1/n_0 = 1/\Omega(n)$$

for $\lambda \geq 2$.

Cost. For each c, since the cost of a union is at most the sum of costs:

$$\sum_{v\in S_c} d(v,c) \ \leq \ \sum_{T\in\mathcal{F}_c}\sum_{v\in T} d(v,c) = \sum_T \mathbf{1}[T \text{ sampled}] \cdot \mathrm{cost}_c(T).$$

Taking expectation and summing over c:

$$\mathbb{E}[\mathrm{ALG}] \leq \sum_{c,T} \mathrm{cost}_c(T) \cdot p_{T,c} \leq \lambda \ln(n) \sum_{c,T} \mathrm{cost}_c(T)\hat{y}_{T,c} = O(\log n) \sum_{c,T} \mathrm{cost}_c(T)\hat{y}_{T,c}.$$

By Markov's inequality, a single run has cost at most twice its expectation with constant probability; therefore, by performing $O(\log n)$ independent runs and returning the minimum-cost feasible solution, we obtain an $O(\log n)$-approximation with high probability. ◀

3.5 Proof of Theorem 3

Now we are ready to complete the proof of Theorem 3.

Proof of Theorem 3. By combining Theorem 9 and Theorem 12, we obtain that if there exists a ρ-approximate density oracle, then there is a polynomial-time algorithm that computes an $O(\rho \log n)$-approximate connected k-median clustering. For general graphs, Lemma 8 provides an $O(\log n)$-approximate density oracle. Substituting $\rho = O(\log n)$, we obtain an $O(\log^2 n)$-approximation. ◀

4 General Version

In this section, we establish bicriteria approximation guarantees for the general version of the connected k-median clustering problem. In particular, we prove Theorem 4, which we restate below.

▶ **Theorem 4.** *Given a vertex set V, a metric $d : V \times V \to \mathbb{R}_{\geq 0}$, an undirected, unweighted connectivity graph $G = (V, E)$ and an integer k, there exists a polynomial-time algorithm that computes an $(O(\log n), O(\log^2 n))$-approximate connected k-median clustering.*

To prove Theorem 4, we follow an approach similar to that used for the assignment version. We first formulate a configuration LP for the general case and compute a near-optimal fractional solution in polynomial time. We then round this solution to obtain the desired bicriteria guarantee. We provide the details in the subsequent sections.

4.1 General Version of the Configuration LP

Primal (Configuration) LP

$$\min \quad \sum_{c \in V} \sum_{T \in \mathcal{T}_c} \mathrm{cost}_c(T)\, y_{T,c} \tag{8}$$

$$\text{s.t.} \quad \sum_{c \in V} \sum_{\substack{T \in \mathcal{T}_c \\ v \in T}} y_{T,c} \geq 1 \qquad \forall v \in V, \tag{9}$$

$$\sum_{c \in V} \sum_{T \in \mathcal{T}_c} y_{T,c} \leq k \tag{10}$$

$$y_{T,c} \geq 0 \qquad \forall c \in V,\ \forall T \in \mathcal{T}_c. \tag{11}$$

4.2 Computing a ρ-Approximate Solution of the Configuration LP for the General Variant

First, we interpret the center-budget constraint as $\sum_{c \in V} \sum_{T \in \mathcal{T}_c} y_{T,c} \leq k$, which bounds the total configuration mass. This serves as a fractional relaxation of the requirement that at most k centers are opened. In particular, the variables $y_{T,c}$ are now defined for all $c \in V$ and all $T \in \mathcal{T}_c$. Let $\mathrm{OPT}_{\mathrm{GCLP}}$ denote the optimal value of this general configuration LP.

▶ **Lemma 13.** *Assume a ρ-approximate rooted minimum-density oracle for arbitrary non-negative node costs and profits. Then, for every $\varepsilon \in (0, 1)$, one can compute in polynomial time a polynomial-support fractional solution $\hat{y}$ satisfying*

$$\sum_{c \in V} \sum_{\substack{T \in \mathcal{T}_c \\ v \in T}} \hat{y}_{T,c} \geq 1 \qquad \forall v \in V, \tag{12}$$

$$\sum_{c \in V} \sum_{T \in \mathcal{T}_c} \hat{y}_{T,c} \leq (1+\varepsilon)k, \tag{13}$$

$$\sum_{c \in V} \sum_{T \in \mathcal{T}_c} \mathrm{cost}_c(T)\hat{y}_{T,c} \leq (1+\varepsilon)\rho \cdot \mathrm{OPT}_{\mathrm{GCLP}}. \tag{14}$$

Moreover, $\hat{y}$ may be chosen with support size polynomial in the input size and $1/\varepsilon$.

Proof. Write the general configuration LP as a mixed packing/covering program with one packing constraint. The covering constraints are indexed by vertices $v \in V$, and the columns are indexed by pairs (c, T) with $c \in V$ and $T \in \mathcal{T}_c$. For such a column, define

$$A_{v,(c,T)} \coloneqq \mathbf{1}[v \in T], \qquad b_{(c,T)} \coloneqq 1, \qquad q_{(c,T)} \coloneqq \mathrm{cost}_c(T).$$

Thus the LP has the form

$$\min \sum_{c,T} q_{(c,T)} y_{T,c} \quad \text{s.t.} \quad Ay \geq \mathbf{1}, \qquad \sum_{c,T} b_{(c,T)} y_{T,c} \leq k, \qquad y \geq 0.$$

We apply the standard oracle-based framework for mixed packing and covering programs, as in Young [47] and Plotkin–Shmoys–Tardos [43]. It suffices to identify the pricing problem required by this framework.

For a guessed objective value τ, the feasibility version is

$$Ay \geq \mathbf{1}, \qquad \sum_{c,T} y_{T,c} \leq k, \qquad \sum_{c,T} \mathrm{cost}_c(T) y_{T,c} \leq \tau, \qquad y \geq 0.$$

In the mixed packing/covering framework, a pricing step is given nonnegative weights p_v on the covering constraints and nonnegative multipliers η, θ for the two packing constraints. It asks for a column minimizing the ratio

$$\frac{\eta \cdot b_{(c,T)} + \theta \cdot q_{(c,T)}}{\sum_{v \in V} p_v A_{v,(c,T)}} = \frac{\eta + \theta\, \mathrm{cost}_c(T)}{\sum_{v \in T} p_v}.$$

For a fixed root c, this is exactly a rooted minimum-density connected-set problem. Indeed, define node costs $w_c^{\eta,\theta}(u) := \theta\, d(u, c) + \eta \cdot \mathbf{1}[u = c]$, and node profits $\alpha_u \coloneqq p_u$. Since every $T \in \mathcal{T}_c$ contains c, we have

$$\sum_{u \in T} w_c^{\eta,\theta}(u) = \theta \sum_{u \in T} d(u, c) + \eta = \theta \mathrm{cost}_c(T) + \eta.$$

Therefore the pricing ratio is precisely $\frac{\sum_{u \in T} w_c^{\eta,\theta}(u)}{\sum_{v \in T} \alpha_v}$. Calling the rooted minimum-density oracle once for each possible root $c \in V$ and taking the best returned column gives a ρ-approximate pricing oracle for the mixed packing/covering instance.

By the standard guarantee for mixed packing and covering with a ρ-approximate pricing oracle, binary search over τ and the above pricing routine compute, in polynomial time, a solution satisfying (12), (13), and (14). The factor $(1 + \varepsilon)$ accounts for the usual accuracy loss in the mixed packing/covering feasibility procedure.

Finally, the algorithm generates only polynomially many columns. Solving the restricted mixed LP over the generated columns and taking a basic feasible solution yields a polynomial-support solution $\hat{y}$ with the same bounds. ◀

▶ **Corollary 14.** *For general connectivity graphs, one can compute in polynomial time a fractional solution $\hat{y}$ satisfying* (12)*,* (13)*, and*

$$\sum_{c \in V} \sum_{T \in \mathcal{T}_c} \mathrm{cost}_c(T) \hat{y}_{T,c} \leq O(\log n) \cdot \mathrm{OPT}_{\mathrm{GCLP}}.$$

Proof. By Lemma 8, the rooted minimum-density oracle has approximation ratio $\rho = O(\log n)$ on general graphs. Applying Lemma 13 with a constant ε gives the claim. ◀

4.3 Rounding the General Version of the Configuration LP

In this subsection, we round the fractional solution obtained from Lemma 13.

▶ **Lemma 15.** *Suppose we are given a polynomial-support feasible solution $\hat{y}$ to (8)–(11) of cost at most Γ. Then there is a randomized polynomial-time algorithm that outputs, with high probability, an overlapping connected clustering using at most $O(k \log n)$ centers of cost $O(\log n) \cdot \Gamma$.*

Proof. Fix a constant $\lambda \geq 2$. Independently for every pair (c, T) in the support of $\hat{y}$, sample the configuration T with probability $p_{T,c} := \min\{1, \lambda \ln n \cdot \hat{y}_{T,c}\}$. For each $c \in V$, let $\mathcal{F}_c$ be the collection of the sampled configurations rooted at c. We open center c if $\mathcal{F}_c \neq \emptyset$, and define its cluster as $S_c := \bigcup_{T \in \mathcal{F}_c} T$. Since every sampled set $T \in \mathcal{T}_c$ contains c and is connected, the union S_c is connected whenever $\mathcal{F}_c \neq \emptyset$.

We first prove coverage. Fix a vertex $v \in V$. By feasibility of $\hat{y}$, $\sum_{c \in V} \sum_{\substack{T \in \mathcal{T}_c \\ v \in T}} \hat{y}_{T,c} \geq 1$. If there exists a pair (c, T) with $v \in T$ and $p_{T,c} = 1$, then v is covered with probability one. Otherwise,

$$\begin{aligned}\Pr[v \text{ is uncovered}] = \prod_{c,T:v \in T} (1 - p_{T,c}) &\leq \exp\Big(- \sum_{c,T:v \in T} p_{T,c} \Big) \\ &= \exp\Big(- \lambda \ln n \sum_{c,T:v \in T} \hat{y}_{T,c} \Big) \leq n^{-\lambda}.\end{aligned}$$

Taking a union bound over all vertices, the probability that some vertex is uncovered is at most $n^{1-\lambda}$, which is at most $1/n$ for $\lambda \geq 2$.

Next, we bound the number of opened centers. Let N be the number of the sampled configurations. Since the number of opened centers is at most N, it suffices to bound N. We have

$$\mathbb{E}[N] = \sum_{c \in V} \sum_{T \in \mathcal{T}_c} p_{T,c} \leq \lambda \ln n \sum_{c \in V} \sum_{T \in \mathcal{T}_c} \hat{y}_{T,c} \leq \lambda(1 + \varepsilon) k \ln n,$$

where the last inequality follows from Constraint (13). By a Chernoff bound, $N = O(k \log n)$ with high probability. Therefore, the algorithm opens at most $O(k \log n)$ centers with high probability.

Finally, we bound the cost. Since the cost of a union is at most the sum of the costs of the sampled configurations,

$$\text{cost}(\{S_c\}) = \sum_{c:\mathcal{F}_c \neq \emptyset} \sum_{v \in S_c} d(v, c) \leq \sum_{c \in V} \sum_{T \in \mathcal{F}_c} \text{cost}_c(T).$$

Taking expectation,

$$\mathbb{E}[\text{cost}(\{S_c\})] \leq \sum_{c \in V} \sum_{T \in \mathcal{T}_c} p_{T,c} \, \text{cost}_c(T) \leq \lambda \ln n \sum_{c \in V} \sum_{T \in \mathcal{T}_c} \text{cost}_c(T) \hat{y}_{T,c} = O(\log n) \cdot \Gamma.$$

By Markov's inequality, a single run has cost at most twice its expectation with constant probability; therefore, by performing $O(\log n)$ independent runs and returning the minimum-cost feasible solution, we obtain an $O(\log n)$-approximation with high probability. This proves the claimed bicriteria guarantee. ◀

4.4 Proof of Theorem 4

Now, we are ready to complete the proof of Theorem 4.

Proof of Theorem 4. Let $\hat{y}$ be the fractional solution returned by Lemma 13 with a fixed constant $\varepsilon \in (0,1)$. Then $\hat{y}$ satisfies the covering constraints and $\sum_{c\in V}\sum_{T\in\mathcal{T}_c} \hat{y}_{T,c} \le (1+\varepsilon)k$, and its cost is at most $(1+\varepsilon)\rho \cdot \mathrm{OPT}_{\mathrm{GCLP}} \le (1+\varepsilon)\rho \cdot \mathrm{OPT}$, where the last inequality follows because the general configuration LP is a relaxation of the non-assignment overlapping connected k-median problem.

Now apply Lemma 15 to $\hat{y}$ with $\Gamma := (1+\varepsilon)\rho \cdot \mathrm{OPT}$. The rounding opens at most $O\big((1+\varepsilon)k\log n\big) = O(k\log n)$ centers with high probability, and its expected cost is at most $O(\log n)\cdot\Gamma = O(\rho\log n)\cdot\mathrm{OPT}$. Thus we obtain an $\big(O(\log n), O(\rho\log n)\big)$ bicriteria approximation.

For general graphs, Lemma 8 gives $\rho = O(\log n)$. Therefore the bicriteria guarantee becomes $\big(O(\log n), O(\log^2 n)\big)$. This proves Theorem 4. ◀

5 Conclusion and Open Problems

In this paper, we developed a configuration-LP framework for the connected k-median problem. Unlike previous cut-based relaxations, our formulation incorporates connectivity directly into the LP variables by working with connected configurations rooted at centers. We combined this viewpoint with covering-LP techniques and rooted minimum-density oracles to obtain approximation guarantees for both the assignment version and a bicriteria variant of the general version. In particular, we obtained an $O(\log^2 n)$-approximation for the assignment version on general graphs, together with an $(O(\log n), O(\log^2 n))$ bicriteria approximation for the general version.

More broadly, our results suggest that configuration LPs provide a natural framework for handling connectivity constraints in clustering problems. By separating the problem into a covering-LP layer and a graph-theoretic rooted density oracle, the framework makes it possible to incorporate stronger oracles for special graph families and related connectivity models. We believe this viewpoint may be useful beyond connected k-median, particularly for other constrained clustering objectives where connectivity plays a central role.

Several interesting open problems remain. The most natural question is whether one can obtain polylogarithmic approximation guarantees for the general version containing exactly k centers. More generally, it would be interesting to better understand the power and limitations of configuration-LP formulations for connectivity-constrained clustering problems. Another promising direction is to design stronger rooted minimum-density oracles for restricted graph families, such as minor-closed graphs. Since the rooted minimum-density problem already admits an $O(\log n)$-approximation in general graphs, improving this guarantee on structured graph families would immediately translate into stronger approximation guarantees within our framework.

References

1 Mohsen Abbasi, Aditya Bhaskara, and Suresh Venkatasubramanian. Fair clustering via equitable group representations. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 504–514, 2021. doi:10.1145/3442188.3445913.

2 Saba Ahmadi, Sainyam Galhotra, Barna Saha, and Roy Schwartz. Fair correlation clustering. *arXiv preprint arXiv:2002.03508*, 2020. arXiv:2002.03508.

3 Sara Ahmadian, Alessandro Epasto, Ravi Kumar, and Mohammad Mahdian. Fair correlation clustering. In *International conference on artificial intelligence and statistics*, pages 4195–4205. PMLR, 2020. URL: http://proceedings.mlr.press/v108/ahmadian20a.html.

4 Sara Ahmadian and Maryam Negahbani. Improved approximation for fair correlation clustering. In *International Conference on Artificial Intelligence and Statistics*, pages 9499–9516. PMLR, 2023. URL: https://proceedings.mlr.press/v206/ahmadian23a.html.

5 Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristic for k-median and facility location problems. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 21–29, 2001. doi:10.1145/380752.380755.

6 Arturs Backurs, Piotr Indyk, Krzysztof Onak, Baruch Schieber, Ali Vakilian, and Tal Wagner. Scalable fair clustering. In *International conference on machine learning*, pages 405–413. PMLR, 2019. URL: http://proceedings.mlr.press/v97/backurs19a.html.

7 MohammadHossein Bateni, MohammadTaghi Hajiaghayi, and Vahid Liaghat. Improved approximation algorithms for (budgeted) node-weighted steiner problems. *SIAM Journal on Computing*, 47(4):1513–1543, 2018.

8 Punam Bedi and Chhavi Sharma. Community detection in social networks. *Wiley interdisciplinary reviews: Data mining and knowledge discovery*, 6(3):115–135, 2016. doi:10.1002/WIDM.1178.

9 Suman Bera, Deeparnab Chakrabarty, Nicolas Flores, and Maryam Negahbani. Fair algorithms for clustering. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

10 Ioana O. Bercea, Martin Groß, Samir Khuller, Aounon Kumar, Clemens Rösner, Daniel R. Schmidt, and Melanie Schmidt. On the Cost of Essentially Fair Clusterings. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2019)*, volume 145 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 18:1–18:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.APPROX-RANDOM.2019.18.

11 Diptarka Chakraborty, Kushagra Chatterjee, Debarati Das, and Tien-Long Nguyen. Generalizing fair clustering to multiple groups: Algorithms and applications. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 19934–19942, 2026. doi:10.1609/AAAI.V40I24.39077.

12 Diptarka Chakraborty, Kushagra Chatterjee, Debarati Das, and Tien-Long Nguyen. A generic framework for fair consensus clustering in streams. *arXiv preprint arXiv:2602.11500*, 2026. doi:10.48550/arXiv.2602.11500.

13 Diptarka Chakraborty, Kushagra Chatterjee, Debarati Das, Tien Long Nguyen, and Romina Nobahari. Towards fair representation: Clustering and consensus. *arXiv preprint arXiv:2506.08673*, 2025. doi:10.48550/arXiv.2506.08673.

14 Moses Charikar and Sudipto Guha. Improved combinatorial algorithms for the facility location and k-median problems. In *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, pages 378–388. IEEE, 1999. doi:10.1109/SFFCS.1999.814609.

15 Moses Charikar, Sudipto Guha, Éva Tardos, and David B Shmoys. A constant-factor approximation algorithm for the k-median problem. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 1–10, 1999.

16 Moses Charikar, Samir Khuller, David M Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In *SODA*, volume 1, pages 642–651, 2001. URL: http://dl.acm.org/citation.cfm?id=365411.365555.

17 Chandra Chekuri, Alina Ene, and Ali Vakilian. Node-weighted network design in planar and minor-closed families of graphs. *ACM Transactions on Algorithms (TALG)*, 17(2):1–25, 2021. doi:10.1145/3447959.

18 Fanglan Chen, Subhodip Biswas, Zhiqian Chen, Shuo Lei, Naren Ramakrishnan, and Chang-Tien Lu. Exploring tradeoffs in automated school redistricting: computational and ethical perspectives. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 15912–15920, 2023. doi:10.1609/AAAI.V37I13.26889.

19 Ke Chen. On k-median clustering in high dimensions. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 1177–1185, 2006. URL: http://dl.acm.org/citation.cfm?id=1109557.1109687.

20 Xingyu Chen, Brandon Fain, Liang Lyu, and Kamesh Munagala. Proportionally fair clustering. In *International conference on machine learning*, pages 1032–1041. PMLR, 2019. URL: http://proceedings.mlr.press/v97/chen19d.html.

21 Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. Fair clustering through fairlets. *Advances in neural information processing systems*, 30, 2017.

22 Vincent Cohen-Addad, Fabrizio Grandoni, Euiwoong Lee, Chris Schwiegelshohn, and Ola Svensson. A (2+ ε)-approximation algorithm for metric k-median. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 615–624, 2025. doi:10.1145/3717823.3718299.

23 Vincent Cohen-Addad, Anupam Gupta, Lunjia Hu, Hoon Oh, and David Saulpic. An improved local search algorithm for k-median. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1556–1612. SIAM, 2022. doi:10.1137/1.9781611977073.65.

24 Zhen Dai, Yury Makarychev, and Ali Vakilian. Fair representation clustering with several protected classes. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 814–823, 2022. doi:10.1145/3531146.3533146.

25 Erik D Demaine, MohammadTaghi Hajiaghayi, and Philip N Klein. Node-weighted steiner tree and group steiner tree in planar graphs. In *International Colloquium on Automata, Languages, and Programming*, pages 328–340. Springer, 2009. doi:10.1007/978-3-642-02927-1_28.

26 Lukas Drexler, Jan Eube, Kelin Luo, Dorian Reineccius, Heiko Röglin, Melanie Schmidt, and Julian Wargalla. Connected k-center and k-diameter clustering. *Algorithmica*, 86(11):3425–3464, 2024. doi:10.1007/S00453-024-01266-9.

27 Seyed Esmaeili, Brian Brubach, Aravind Srinivasan, and John Dickerson. Fair clustering under a bounded cost. *Advances in Neural Information Processing Systems*, 34:14345–14357, 2021. URL: https://proceedings.neurips.cc/paper/2021/hash/781877bda0783aac5f1cf765c128b437-Abstract.html.

28 Seyed Esmaeili, Brian Brubach, Leonidas Tsepenekas, and John Dickerson. Probabilistic fair clustering. *Advances in Neural Information Processing Systems*, 33:12743–12755, 2020.

29 Jan Eube, Kelin Luo, Dorian Reineccius, Heiko Röglin, and Melanie Schmidt. Connected k-median with disjoint and non-disjoint clusters. In *Proceedings of the 33rd Annual European Symposium on Algorithms (ESA)*, LIPIcs, 2025. doi:10.4230/LIPIcs.ESA.2025.63.

30 Hossein Fani and Ebrahim Bagheri. Community detection in social networks. *Encyclopedia with semantic computing and robotic intelligence*, 1(01):1630001, 2017.

31 Rong Ge, Martin Ester, Byron J Gao, Zengjian Hu, Binay Bhattacharya, and Boaz Ben-Moshe. Joint cluster analysis of attribute data and relationship data: The connected k-center problem, algorithms and applications. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2(2):1–35, 2008. doi:10.1145/1376815.1376816.

32 Mehrdad Ghadiri, Samira Samadi, and Santosh Vempala. Socially fair k-means clustering. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 438–448, 2021. doi:10.1145/3442188.3445906.

33 Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical computer science*, 38:293–306, 1985. doi:10.1016/0304-3975(85)90224-5.

34 Neelima Gupta, Aditya Pancholi, and Yogish Sabharwal. Clustering with internal connectedness. In *International Workshop on Algorithms and Computation*, pages 158–169. Springer, 2011. doi:10.1007/978-3-642-19094-0_17.

35 Dorit S Hochbaum and David B Shmoys. A best possible heuristic for the k-center problem. *Mathematics of operations research*, 10(2):180–184, 1985. doi:10.1287/MOOR.10.2.180.

36 Wen-Lian Hsu and George L Nemhauser. Easy and hard bottleneck location problems. *Discrete Applied Mathematics*, 1(3):209–215, 1979. doi:10.1016/0166-218X(79)90044-1.

37 Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 731–740, 2002. doi:10.1145/509907.510012.

38 Christopher Jung, Sampath Kannan, and Neil Lutz. Service in your neighborhood: Fairness in center location. *Foundations of Responsible Computing (FORC)*, 2020.

39 Samir Khuller and Yoram J Sussmann. The capacitated k-center problem. *SIAM Journal on Discrete Mathematics*, 13(3):403–418, 2000. doi:10.1137/S0895480197329776.

40 Shi Li. Approximating capacitated k-median with k open facilities. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 786–796. SIAM, 2016.

41 Sepideh Mahabadi and Ali Vakilian. Individual fairness for k-clustering. In *International conference on machine learning*, pages 6586–6596. PMLR, 2020. URL: http://proceedings.mlr.press/v119/mahabadi20a.html.

42 Yury Makarychev and Ali Vakilian. Approximation algorithms for socially fair clustering. In *Conference on Learning Theory*, pages 3246–3264. PMLR, 2021. URL: http://proceedings.mlr.press/v134/makarychev21a.html.

43 S. A. Plotkin, D. B. Shmoys, and É. Tardos. Fast approximation algorithms for fractional packing and covering problems. In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science (FOCS)*, 1991. doi:10.1109/SFCS.1991.185411.

44 Eklavya Sharma. An approximation algorithm for covering linear programs and its application to bin-packing. *arXiv preprint arXiv:2011.11268*, 2020. arXiv:2011.11268.

45 Ali Vakilian and Mustafa Yalciner. Improved approximation algorithms for individually fair clustering. In *International conference on artificial intelligence and statistics*, pages 8758–8779. PMLR, 2022.

46 Hamidreza Validi, Austin Buchanan, and Eugene Lykhovyd. Imposing contiguity constraints in political districting models. *Operations Research*, 70(2):867–892, 2022. doi:10.1287/OPRE.2021.2141.

47 Neal E Young. Sequential and parallel algorithms for mixed packing and covering. In *Proceedings 42nd IEEE symposium on foundations of computer science*, pages 538–546. IEEE, 2001. doi:10.1109/SFCS.2001.959930.

A Proof of Lemma 5

Proof. Fix a feasible overlapping solution $(S_c)_{c\in C}$. By feasibility, for every $c \in C$, we have $c \in S_c$ and $G[S_c]$ is connected. Hence $S_c \in \mathcal{T}_c$ for every center c. Define a configuration-LP solution y by

$$y_{T,c} := \begin{cases} 1, & \text{if } T = S_c, \\ 0, & \text{otherwise.} \end{cases}$$

We verify feasibility. First, for every $v \in V \setminus C$, the overlapping solution satisfies $\bigcup_{c\in C} S_c = V$, so there exists at least one center c such that $v \in S_c$. Therefore, $\sum_{c\in C} \sum_{\substack{T\in\mathcal{T}_c \\ v\in T}} y_{T,c} \geq 1$, and thus the covering constraints (2) hold for all $v \in V \setminus C$. Second, by construction, all variables satisfy $y_{T,c} \geq 0$, so (3) also holds. Finally, the objective value of this LP solution is

$$\sum_{c\in C} \sum_{T\in\mathcal{T}_c} \mathrm{cost}_c(T)\, y_{T,c} = \sum_{c\in C} \mathrm{cost}_c(S_c) = \sum_{c\in C} \sum_{v\in S_c} d(v,c),$$

which is exactly the cost of the given integral solution.

Since every feasible solution to the assignment variant of connected k-median with overlapping clusters induces a feasible solution to the configuration LP with the same cost, the optimal value of the configuration LP is at most OPT, the optimal cost of connected k-median with overlapping clusters for the prescribed center set C. ◀

B Proof of Lemma 8

Proof. Write $w(T) := \sum_{v\in T} w(v)$ and $\alpha(T) := \sum_{v\in T} \alpha_v$. Let T^* be an optimal density solution with $\lambda^* := w(T^*)/\alpha(T^*)$.

We use the bicriteria guarantee of Bateni et al. [7, Theorem 3] for the *rooted budgeted node-weighted Steiner tree problem*. In this problem, we are given a graph $G = (V, E)$, a designated root $c \in V$, a non-negative node-cost function $w : V \to \mathbb{R}_{\geq 0}$, a non-negative node-profit function $\alpha : V \to \mathbb{R}_{\geq 0}$, and a budget $B \geq 0$. The goal is to find a connected tree $T \subseteq G$ containing c whose total node cost is at most B and whose total profit $\alpha(T) := \sum_{v\in T} \alpha(v)$ is as large as possible. Let

$$\alpha^*(B) := \max\{\alpha(T) : T \ni c \text{ is connected and } w(T) \leq B\} \text{ where } w(T) := \sum_{v\in T} w(v).$$

Then, for any budget B and any $\varepsilon > 0$, there is a polynomial-time algorithm that returns a connected tree $T_B \ni c$ such that $w(T_B) \leq (1+\varepsilon)B$ and $\alpha(T_B) \geq \frac{\varepsilon}{O(\log n)} \cdot \alpha^*(B)$. In other words, the algorithm may violate the budget by a factor of at most $1+\varepsilon$, while guaranteeing an $O(\log n/\varepsilon)$-approximation to the maximum achievable profit under budget B.

Set $W_{\max} := w(V)$ and try the $O(\log W_{\max})$ budgets $B \in \{1, 2, 4, \ldots, 2^{\lceil \log_2 W_{\max} \rceil}\}$. There exists $B^\star$ in this list with $w(T^*) \leq B^\star \leq 2w(T^*)$. Apply the guarantee of Bateni et al. [7] with $\varepsilon := 1$. Since $w(T^*) \leq B^\star$, we have $\alpha^*(B^\star) \geq \alpha(T^*)$. The algorithm returns $T_{B^\star} \ni c$ with $w(T_{B^\star}) \leq 2B^\star \leq 4w(T^*)$ and $\alpha(T_{B^\star}) \geq \alpha(T^*)/O(\log n)$. Therefore,

$$\operatorname{dens}(T_{B^\star}) = \frac{w(T_{B^\star})}{\alpha(T_{B^\star})} \leq \frac{4\,w(T^*)}{\alpha(T^*)/O(\log n)} = O(\log n) \cdot \lambda^*.$$

Returning $\arg\min_B \operatorname{dens}(T_B)$ over all tried budgets yields an $O(\log n)$-approximate density oracle in polynomial time. ◀

C Proof of Theorem 10

Proof. The covering LP $\min\{c^\top y : Ay \geq \mathbf{1},\ y \geq 0\}$ has (in our usage) exponentially many columns, so we cannot solve it directly; we only have oracle access to an approximately-best column for any given constraint weighting. The proof has three parts.

1. We first observe that the α-approximate *pricing* oracle we are given is indeed also an η-weak *index-finding* oracle ($\eta = 1/\alpha$) for a natural quantity $D_j(w)$, the "coverage per unit weight" of column j.
2. We then show that an index-finding oracle over columns is the same thing as a point-finding oracle over a certain simplex-like polytope P_r (all vectors of cost exactly r). This is the step that lets us invoke a known black-box algorithm for fractional covering, which expects a point-finding oracle, not a column oracle.
3. The black-box algorithm, for a *fixed* guess r of the optimal cost, either certifies cost r suffices (up to a small violation) or fails. We do not know OPT in advance, so we do a binary search over r; two monotonicity facts about this success/failure signal pin down $r \approx \mathrm{OPT}$ and let us convert the near-feasible solution found at the end into a genuinely feasible one, at a controlled extra cost.

In what follows, we elaborate on each of these steps.

Part 1: from pricing to index-finding. For a column j and weight vector $w \in \mathbb{R}^m_{\geq 0}$, let $D_j(w) := \frac{w^\top A_{\cdot j}}{c_j} = \frac{\sum_i A_{ij} w_i}{c_j}$. Think of $D_j(w)$ as how much weighted coverage column j buys you per unit of its cost. An oracle is an *η-weak index-finding oracle* if, given $w \geq 0$, it returns some j with $D_j(w) \geq \eta \max_\ell D_\ell(w)$, in other words a column that is within a factor $\eta \leq 1$ of the best possible coverage-per-cost.

The assumed pricing oracle minimizes the *reciprocal* quantity $c_j/(w^\top A_{\cdot j}) = 1/D_j(w)$ to within a factor α: If $\frac{c_j}{w^\top A_{\cdot j}} \leq \alpha \cdot \min_{j'} \frac{c_{j'}}{w^\top A_{\cdot j'}}$, then taking reciprocals (all quantities are positive since $c > 0$; the degenerate case $\max_\ell D_\ell(w) = 0$, i.e. when w is orthogonal to every column, makes the index-finding requirement vacuous and is set aside), then

$$D_j(w) \geq \frac{1}{\alpha} \max_\ell D_\ell(w).$$

So the pricing oracle *is* an η-weak index-finding oracle with $\eta := 1/\alpha$.

Part 2: from index-finding to point-finding on P_r. Fix a target cost $r \geq 0$ and consider the cost-r slice of the nonnegative orthant, $P_r := \{y \in \mathbb{R}^N_{\geq 0} : c^\top y = r\}$. We introduce this set because known fractional-covering algorithms are stated abstractly for a polytope P together with a *point-finding* oracle (return an approximately-best point of P for a given linear objective), and we want to feed our column oracle into one of them. The set P_r is exactly the right polytope: every $y \in P_r$ can be written as $y = \sum_{j=1}^N \frac{c_j y_j}{r} \cdot \frac{r}{c_j} e_j$, and the coefficients $c_j y_j / r$ are nonnegative and sum to 1 (since $\sum_j c_j y_j = c^\top y = r$), so y is a convex combination of the points $\{(r/c_j)e_j : j \in [N]\}$. In other words, P_r is the simplex whose vertices are, for each column j, spend the entire budget r on column j alone. Maximizing a linear function over a simplex is achieved at a vertex, so for any weight vector $w \geq 0$,

$$\max_{y \in P_r} w^\top A y = \max_{j \in [N]} w^\top A\Big(\frac{r}{c_j} e_j\Big) = r \cdot \max_j D_j(w).$$

This identity is the crux of this part: finding a good *point* of P_r for objective $w^\top A(\cdot)$ is literally the same problem as finding a good *column* for $D_j(w)$, just rescaled by r. So our η-weak index-finding oracle, applied to w, returns a column j with $D_j(w) \geq \eta \max_\ell D_\ell(w)$; setting $y := (r/c_j)e_j \in P_r$ gives

$$w^\top A y = r D_j(w) \geq \eta \cdot r \max_\ell D_\ell(w) = \eta \max_{y' \in P_r} w^\top A y',$$

In other words, y is an η-weak maximizer of $w^\top A y$ over P_r. We have therefore converted our column oracle into an η-weak point-finding oracle for P_r, for every r simultaneously (the construction above does not depend on r except through the final rescaling).

We also record the *width* of this setup, since the black-box algorithm's running time depends on it: by the same vertex computation,

$$\mathrm{width}(A, \mathbf{1}, P_r) := \max_{y \in P_r} \max_i (Ay)_i = \max_{i,j} A_{ij} \cdot \frac{r}{c_j} = r \max_{i,j} \frac{A_{ij}}{c_j}.$$

Using any known bound $q \geq \mathrm{OPT}$ and the given $W \geq q \max_{i,j} A_{ij}/c_j$, we get, for every $r \leq q$,

$$\mathrm{width}(A, \mathbf{1}, P_r) = r \max_{i,j} \frac{A_{ij}}{c_j} \leq q \max_{i,j} \frac{A_{ij}}{c_j} \leq W.$$

Part 3: black-box covering, then binary search on r. We now invoke, as a black box, the standard guarantee for fractional covering with an approximate point-finding oracle (Plotkin–Shmoys–Tardos; we use the clean restatement of Sharma [44], Theorems 3–4): given the η-weak point-finding oracle for P_r from Part 2 and the width bound W from Part 2, and setting $\mu := \frac{\eta}{1+\epsilon}$, the algorithm either correctly reports $\{y \in P_r : Ay \geq \mathbf{1}\}$ is empty or returns some $y \in P_r$ with $Ay \geq \mu\mathbf{1}$ (i.e. every covering constraint is satisfied up to a $(1-\mu)$ shortfall) using a number of oracle calls that yields support size $|\mathrm{supp}(y)| = \widetilde{O}(mW/(\eta\epsilon^3))$ when the oracle always returns a single vertex (which ours does, by construction in Part 2). Call this routine on input r and write $g(r) = 1$ if it succeeds (returns such a y) and $g(r) = 0$ if it reports infeasibility.

The routine above only tells us, for a *given* r, whether spending exactly r can nearly cover everything. We do not know OPT, so we search for the smallest r at which $g(r)$ flips from 0 to 1. The bisection argument has two parts:

(i) $r \geq \mathrm{OPT} \implies g(r) = 1$. Let y^* be an optimal solution of the original covering LP, so $c^\top y^* = \mathrm{OPT}$ and $Ay^* \geq \mathbf{1}$. Rescaling, $\frac{r}{\mathrm{OPT}} y^* \in P_r$ (it has cost exactly r) and $A\big(\frac{r}{\mathrm{OPT}} y^*\big) = \frac{r}{\mathrm{OPT}} Ay^* \geq \frac{r}{\mathrm{OPT}}\mathbf{1} \geq \mathbf{1}$ since $r \geq \mathrm{OPT}$. So $\{y \in P_r : Ay \geq \mathbf{1}\}$ is genuinely nonempty, and since it is nonempty (and in particular $Ay \geq \mathbf{1} \geq \mu\mathbf{1}$ trivially, as $\mu \leq 1$), the routine cannot correctly report infeasibility; hence $g(r) = 1$.

(ii) $r < \mu \cdot \mathrm{OPT} \implies g(r) = 0$. Suppose instead $g(r) = 1$, i.e. the routine returns $y \in P_r$ with $Ay \geq \mu\mathbf{1}$. Rescale: $\hat{y} := y/\mu$ satisfies $A\hat{y} = Ay/\mu \geq \mathbf{1}$, so $\hat{y}$ is feasible for the *original* covering LP, with cost $c^\top \hat{y} = (c^\top y)/\mu = r/\mu < \mathrm{OPT}$ (using $r < \mu \cdot \mathrm{OPT}$). This contradicts optimality of OPT. So $g(r) = 1$ is impossible, i.e. $g(r) = 0$.

Then, we initialize $\ell = 0$, $u = q$ (both satisfy $g(\ell) = 0, g(u) = 1$, using fact (ii) at $\ell = 0$ whenever $\mathrm{OPT} > 0$), and repeatedly query the midpoint $r = (\ell + u)/2$: if $g(r) = 0$ set $\ell := r$, otherwise set $u := r$ and record the returned vector. Stop once $u \leq (1+\delta)\ell$ for $\delta := \varepsilon^2/(1+\varepsilon)$; this takes $O(\log(q/(\delta \cdot (\text{smallest gap}))))$ iterations, polynomial in the input size.

Finally, at termination $g(\ell) = 0$ forces $\ell < \mathrm{OPT}$ (else fact (i) would give $g(\ell) = 1$), so $u \leq (1+\delta)\ell < (1+\delta)\,\mathrm{OPT}$. Let $y^{(u)} \in P_u$ be the vector recorded at the last successful call ($g(u) = 1$), so $Ay^{(u)} \geq \mu\mathbf{1}$ and $c^\top y^{(u)} = u$. Define $\hat{y} := \frac{1}{\mu} y^{(u)}$. Exactly as in fact (ii)'s rescaling, $A\hat{y} \geq \mathbf{1}$, so $\hat{y}$ is feasible for the original covering LP. Its cost is $c^\top \hat{y} = \frac{u}{\mu} \leq \frac{(1+\delta)}{\mu}\mathrm{OPT}$. Substituting $\mu = \eta/(1+\varepsilon)$ and $\delta = \varepsilon^2/(1+\varepsilon)$:

$$\frac{1+\delta}{\mu} \;=\; \Big(1 + \frac{\varepsilon^2}{1+\varepsilon}\Big) \cdot \frac{1+\varepsilon}{\eta} = \frac{1+\varepsilon+\varepsilon^2}{\eta} = \alpha(1+\varepsilon+\varepsilon^2),$$

using $\eta = 1/\alpha$ from Part 1. This gives the claimed cost bound $c^\top \hat{y} \leq \alpha(1+\varepsilon+\varepsilon^2)\mathrm{OPT}$. Rescaling by $1/\mu$ does not change the support, so

$$|\mathrm{support}(\hat{y})| = |\mathrm{support}(y^{(u)})| = \widetilde{O}\Big(\frac{mW}{\eta\varepsilon^3}\Big) = \widetilde{O}\Big(\frac{mW\alpha}{\varepsilon^3}\Big),$$

matching the statement of the theorem.

Runtime. Each bisection step invokes the black-box covering routine once, which in turn calls our point-finding oracle (equivalently, the original pricing oracle, once per iteration of that routine) $\mathrm{poly}(m, W, 1/\varepsilon)$ times; the number of bisection steps is $\mathrm{poly}(\log(q/\mathrm{OPT}), \log(1/\delta))$. All oracle calls and arithmetic operations are polynomial whenever q, W, and the input encoding lengths are polynomially bounded, giving the stated overall polynomial running time. ◀

Approximation Algorithms for Matroidal Prerequisite Systems

Robert P. Streit
Dept. of Electrical & Computer Engineering, The University of Texas at Austin, TX, USA

Vijay K. Garg
Dept. of Electrical & Computer Engineering, The University of Texas at Austin, TX, USA

Abstract

Optimal selections in a decision process are often constrained by prerequisites. However, such prerequisites can encode functional rather than literal dependencies, so a required dependency may be supplied by one or several interacting alternatives. We introduce *matroidal prerequisite systems (MPS)*, a combinatorial constraint structure where a poset specifies prerequisites while a matroid determines when those prerequisites have been satisfied by its span. This creates an order-sensitive notion of feasibility over words, where feasible words are associated with independent sets, while dependencies may be fulfilled through substitutable functionality.

Our main contribution is approximation algorithms for nonnegative additive maximization and monotone submodular maximization over the feasible words of an MPS. The guarantees are determined by two structural parameters: the maximum matroid rank Δ of a principal ideal in the poset and the maximum matroid connectivity $\lambda_{\max}$. These measure the distance an MPS is from encoding a matroid or a poset antimatroid, respectively, both of which are generalized by an MPS. For additive maximization, we obtain deterministic Δ- and $(1+\lambda_{\max})$-approximation algorithms. By extending these techniques, we obtain efficient deterministic $(2+\lambda_{\max})$-approximation and randomized $(\Delta^2 \cdot (1-1/e-\delta)^{-1})$-approximation algorithms for all $\delta > 0$ for submodular maximization. The algorithm design and analysis use the theory of polymatroid greedoids, via a cryptomorphism we prove between an MPS and a *strong polymatroid greedoid*. Finally, a reduction from densest k-subgraph shows it is not possible to efficiently compute a $\min\{\Delta, \lambda_{\max}\}^{o(1)}$-approximation to nonnegative additive maximization over the feasible words of an MPS under the Gap Exponential Time Hypothesis. Thus, an MPS provides a tractable, but provably nontrivial, framework for combinatorial optimization with interacting prerequisites, independence, and substitution.

2012 ACM Subject Classification Mathematics of computing → Combinatorial optimization; Mathematics of computing → Matroids and greedoids; Theory of computation → Approximation algorithms analysis

Keywords and phrases matroids, posets, polymatroid greedoids, submodular maximization

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.23

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2607.08151

Funding *Vijay K. Garg*: Partially supported by the Cullen Trust for Higher Education Endowed Professorship.

1 Introduction

In a discrete decision process, some decisions are not meaningful, feasible, or useful until other functionality is already present. This is perhaps most obvious in task scheduling problems, where dependencies impose constraints on what can and cannot be completed. However, such dependencies are rarely literal. Some project task may require not an earlier task itself, but rather some interface or information that earlier tasks are meant to provide. These required


© Robert P. Streit and Vijay K. Garg;
licensed under Creative Commons License CC-BY 4.0



functionalities may be satisfied by alternative selections of one or many interacting decisions. Yet, existing prerequisite models (e.g. [38, 19, 41, 31, 20]) typically require prerequisites to be selected explicitly; i.e. if y is a prerequisite of z, then z can only be chosen after y. So, there is no awareness of the overlap or substitutable nature of the decision functionalities.

Matroids encode such a substitution structure. A *matroid* $\mathcal{M}$ is a nonempty set family, whose members are called *independent*, such that every subset of an independent set is independent and for all independent X and Y with $|Y| > |X|$ there exists $z \in Y \setminus X$ such that $X + z$ is independent. The *rank* $r_{\mathcal{M}}(X)$ of an arbitrary set X is the size of a maximal independent set it contains, while the *span* $\sigma_{\mathcal{M}}(X)$ is the union of X with the elements which do not increase its rank. Now, if feasible sequences of decisions $x_1 \dots x_\ell$ are always such that $\{x_1, \dots, x_\ell\}$ is independent, then the span $\sigma_{\mathcal{M}}(\{x_1, \dots, x_\ell\})$ can encode the functionalities covered by $x_1 \dots x_\ell$. So, two decisions $y, z \notin \sigma_{\mathcal{M}}(\{x_1, \dots, x_\ell\})$ which are not made redundant by these functionalities (in the sense they are spanned by $\{x_1 \dots x_\ell\}$) are $\{x_1, \dots, x_\ell\}$-*substitutable* if and only if $\sigma_{\mathcal{M}}(\{x_1, \dots, x_\ell, y\}) = \sigma_{\mathcal{M}}(\{x_1, \dots, x_\ell, z\})$. This is, given what is already covered by $x_1 \dots x_\ell$, any new functionality added by y or z leads to equivalent coverage. Matroids hold a central place in discrete optimization because their structure models many problems and gives algorithms for many settings (see [40, 10, 35, 5, 14, 45]).

In this work, we study the optimization problems that arise when prerequisite and matroidal substitution structure are forced to interact. Our reasons are both modeling based and algorithmic. Many applications lead to prerequisite structure defined via substitutable functionalities rather than irreplaceable objects, and we would like to capture this. We will also see the resulting constraint systems form a structured class weaker than matroids that still admits approximation algorithms for additive and submodular maximization. To that end, we propose the *matroidal prerequisite system (MPS)*, which is a simple language (words without repeating letters) over a finite alphabet Σ defined by an interaction between a matroid $(\Sigma, \mathcal{M})$ and poset $(\Sigma, \rightsquigarrow)$. The words $y_1 \dots y_\ell$ in the language, denoted by $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$, are such that their letters always form independent sets, and each y_i is such that all prerequisites $x \rightsquigarrow y_i$ are either spanned by the prefix $\{y_1, \dots, y_{i-1}\}$ in the matroid or x and y_i are $\{y_1, \dots, y_{i-1}\}$-substitutable. This, along with a compatibility condition stating that letters spanned by feasible words must be such that their prerequisites are also spanned, defines an MPS. The upshot is the poset and matroid retain their usual roles, but no longer act independently. Specifically, the poset specifies which decisions require earlier prerequisite coverage, and the matroid determines when that prerequisite coverage has been supplied via its span while continuing to enforce nonredundancy via independence. An MPS generalizes matroids, as $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is the permutations of independent sets if the ordering relation is empty, as well as *poset antimatroids* (the language of linear extensions of a poset) as $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is the linear extensions of $(\Sigma, \rightsquigarrow)$ if $\mathcal{M}$ is free (all sets are independent).

Our main contributions are approximation algorithms for maximization over the words of an MPS of nonnegative additive and monotone submodular functions. Our guarantees are given in terms of structural parameters measuring the distance of an MPS from being a matroid or poset antimatroid. First is the largest matroid rank of a principal ideal,

$$\Delta \triangleq \max_{z \in \Sigma} r_{\mathcal{M}}(\downarrow z), \quad \text{where } \downarrow z \triangleq \{y \in \Sigma \mid y \rightsquigarrow z\} \cup \{z\}. \tag{1}$$

If the MPS is a matroid (i.e. "$\rightsquigarrow$" is empty) then $\Delta = 1$, and as the prerequisites become more complex Δ will increase. The second parameter is defined by the largest value taken by the (Tutte) connectivity function λ (see [47, 48, 37]) of the matroid,

$$\lambda_{\max} \triangleq \max_{X \subseteq \Sigma} \lambda(X), \quad \text{where } \lambda(X) \triangleq r_{\mathcal{M}}(X) + r_{\mathcal{M}}(\Sigma \setminus X) - r_{\mathcal{M}}(\Sigma). \tag{2}$$

The connectivity $\lambda(X)$ computes how much the complement $\Sigma \setminus X$ can cover X in its span, in terms of rank. Equivalently, $\lambda(X)$ measures how much of X can be substituted with its complement. So, $\lambda_{\max}$ measures how complex the substitutes structure is. Note $\lambda_{\max} \geq 0$. For free matroids where every set is independent, we see $\lambda_{\max} = 0$. So, $\lambda_{\max} = 0$ when the MPS defines a poset antimatroid and increases as the substitutes structure becomes more complex. This leads to our first main result.

▶ **Main Result 1** (Theorems 12, 13, 14, and 15)**.** *Let $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ be a matroidal prerequisite system. Assume we are given a matroid rank oracle.*

1. *There exists an efficient deterministic $\min\{\Delta, 1 + \lambda_{\max}\}$-approximation algorithm for nonnegative additive maximization over the feasible words of $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$,*
2. *There exists an efficient deterministic $(2 + \lambda_{\max})$-approximation and, for every $\delta > 0$, an efficient randomized $(\Delta^2 \cdot (1 - 1/e - \delta)^{-1})$-approximation algorithm for monotone submodular maximization over the feasible words of $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$.*

Observe, our methods achieve exact solutions for additive maximization whenever the MPS is a matroid or poset antimatroid. These are the two "easy" cases, and the approximation smoothly degrades as we drift towards a more complex MPS structure (as Δ and $\lambda_{\max}$ will increase). For submodular maximization we achieve essentially the best possible approximation [34] when the MPS is a matroid. For poset antimatroids, the problem becomes trivialized as the entirety of Σ can always be permuted into a linear extension, so outputting this permutation always leads to an exact solution even for monotone submodular objectives. But, our guarantee would give a 2-approximation. Whether it is possible to improve this to smoothly interpolate between this scenario and more complex MPS structures (like in the additive case) is something we leave to future work. Finally, we note that the expected $(\Delta^2 \cdot (1 - 1/e - \delta)^{-1})$-approximation guarantee comes from using the continuous greedy algorithm [5] as a black-box subroutine. We present our methods with this chosen subroutine because it is the most recognizable to the widest audience, but one could replace this with any of [45, 14, 4, 7] to obtain various improvements.

Our algorithmic techniques make use of *strong polymatroid greedoids* recently introduced in [44]. Specifically, a polymatroid greedoid is a greedoid whose feasibility structure can be encoded by a polymatroid rank function called a *representation*. Strong polymatroid greedoids are a subclass admitting certain closure operators and a canonical representation called *the greatest representation*. We prove a cryptomorphism in Theorem 9 showing a greedoid is a strong polymatroid greedoid if and only if it can be encoded as an MPS. This allows us to use the greatest representation in our algorithms after verifying this function can be evaluated efficiently using a matroid rank oracle. In particular, Section 5 presents algorithms based off of solving a relaxation over the base polytope of the greatest representation, while Section 6 uses the greatest representation to create modified greedy algorithms. The correctness will follow from Lemma 11, which discusses how to iteratively "fill-in" sets of letters contained in feasible words to converge to a full feasible word in a strong polymatroid greedoid.

To complement the algorithmic results, we also investigate the hardness of nonnegative additive maximization. We give an approximation-preserving reduction from Densest k-Subgraph. By [29], this shows our algorithmic guarantees are meaningful in that exponential improvements are not possible under the Gap Exponential Time Hypothesis (see [30, 9, 6]).

▶ **Main Result 2.** *Under the Gap Exponential Time Hypothesis, there is no $\min\{\Delta, \lambda_{\max}\}^{o(1)}$-approximation algorithm for additive maximization over a matroidal prerequisite system.*

Our reduction uses uniform matroids, so we note that this hardness applies even to an MPS defined with a uniform matroid. While Main Result 2 shows constant or logarithmic-approximation algorithms in these parameters are unlikely, the result doesn't comment on sublinear approximations in their entirety. We leave the resolution of this gap to future work.

Overview. We begin by surveying related work and making comparisons in Section 2. Then, in Section 3 we overview some background. Section 4 will introduce the MPS. Section 4.1 examines the greedoid theory of an MPS, and proves a cryptomorphism between the class of MPS and strong polymatroid greedoids from [44] in Theorem 9. Section 4.2 shows that the greatest representation of a MPS can be evaluated efficiently via a matroid rank query, while Section 4.3 proves that every sequence of letters such that the marginal return of each letter against its prefix in the greatest representation is greater than zero can be "filled" into a feasible word by repeated letter insertions.

In Section 5 we design approximation algorithms with guarantees varying in Δ. Section 5.1 gives a method for additive maximization which is based on solving a relaxation defined by the greatest representation and projecting the solution onto a feasible word using Lemma 11. This is Algorithm 1, which Theorem 12 shows is a Δ-approximation algorithm. Then, in Section 5.2 we obtain a method for monotone submodular maximization by applying the continuous greedy algorithm [5] to the *natural matroid* construction [16, 3] of the greatest representation, and using the solution to create an additive underestimation which is fed into Algorithm 3. The analysis in Theorem 13 shows this process leads to an expected approximation at most a Δ^2 factor worse than that of [5] for submodular maximization over matroids. In Section 6, we design approximation algorithms with guarantees varying in $\lambda_{\max}$. Section 6.1 begins by examining algorithms for additive objectives, and presents a modified greedy algorithm (Algorithm 3) using the greatest representation to adjust the perceived value of available letters in each round, and the closure operator of strong polymatroid greedoids to determine where to insert selected letters into a working solution. Theorem 14 shows the resulting solution is a $(1 + \lambda_{\max})$-approximation. Then, Section 6.2 extends the technique to achieve a $(2 + \lambda_{\max})$-approximation for submodular objective functions. Finally, we conclude by proving Main Result 2 in Section 7.

2 Related Work

Our MPS is in many ways a relaxation of matroids for designing approximation algorithms in more general settings. Other approaches have been made in this spirit. See for example p-systems [35], k-intersection [21, 27], k-extendible [32], and k-exchange [13] systems. All come with approximate guarantees decaying in the "distance" from a matroid, like the Δ parameter used in Main Result 1. But, a key difference is our feasibility model is order-sensitive. This is to say, where previous works maintain a subset-closed set family, our approach injects a notion of prerequisites using a poset that destroys such structure. This is why we interpret our MPS as a greedoid [25], as this theory accommodates order-sensitive feasibility.

On that note, there do exist works mixing prerequisite structure with other combinatorial constraint systems. A notable example is precedence constrained knapsack variants [41, 19, 20] and maximum closure [38, 31] problems. However, the prerequisite constraints of these problems require a "complete" closure, in the sense that an element can only be selected once *all* of its prerequisites are present. In contrast, we use a matroid to encode notion of substitution or redundancy. Along these lines, AND/OR scheduling problems [33] use Boolean formulae to allow for alternatives through disjunction. However, in [33] the formulae are explicit inputs, while our delegation to the matroid induces a stateful and implicit notion of substitutability which varies over the decision process.

Our work uses greedoids, a structure introduced by Korte and Lovász to generalize matroids by removing the definitional dependence on subset-closed set families [22, 25, 2]. For linear optimization, [23, 15] give a necessary and sufficient called *strong exchange* for

the correctness of the greedy algorithm in greedoids, which was later generalized to the idea of a *matroid embedding* in [17]. In Section 4 we will see that the greedy algorithm fails to solve additive maximization on an MPS, and so these results do not apply to our setting. Without strong exchange, there do exist certain characterizations on the types of objective functions which the greedy algorithm can correctly solve [25, 46, 12], but they are sensitive to the greedoid structure and quite restrictive. Polymatroid greedoids are introduced in [24], and recently studied in [44]. This class is defined by an interaction between a polymatroid and a greedoid which allows for the use of submodular analysis (which is typically not fully available in the greedoid setting, unlike matroids). The latter work [44] introduces *strong polymatroid greedoids*, which give special properties we use for the design and analysis of our algorithms after showing an MPS is cryptomorphically equivalent to such a greedoid.

3 Preliminaries

For any set X, let $X!$ give its permutations, and $\chi_X \in \{0,1\}^\Sigma$ be its characteristic vector. We often iterate over words written like $y_1 \dots y_{i-1}$, so we are implicitly writing $y_1 \dots y_0$ when $i = 1$. Let $y_1 \dots y_0 = \epsilon$, where ϵ is the empty word, and $\{y_1, \dots, y_0\} = \varnothing$. Finally, for $v \in \mathbb{R}^\Sigma$, the *support* $\operatorname{supp}(v) \triangleq \{y \in \Sigma \mid v(y) \neq 0\}$ is the set of labels for the nonzero coordinates in v.

Submodularity. An *additive function* is a set function defined by assigning each letter a weight via $w : \Sigma \to \mathbb{R}$ and mapping sets to the sum of the weights of letters they contain, like $X \mapsto \sum_{y \in X} w(y)$. A *submodular function* $f : 2^\Sigma \to \mathbb{R}$ is a set function satisfying the *law of diminishing returns* defined as $X \subseteq Y$ implies $(f/X)(z) \geq (f/Y)(z)$ for all $z \in \Sigma$. where $(f/X)(z) \triangleq f(X + z) - f(X)$ is the evaluation of the *contraction of f by X* at the letter z. We adopt the convention that $(f/X)(z) = 0$ whenever $z \in X$. Note, if $X = \{y_1, \dots, y_\ell\}$, then $f(X) = \sum_{i=1}^{\ell} (f/\{y_1, \dots, y_{i-1}\})(y_i)$. And so, submodular functions generalize additive functions. Nonnegative additive and monotone (i.e. $X \subseteq Y$ implies $f(X) \leq f(Y)$) submodular functions are the objective functions we consider. We will assume that f is *normalized*, i.e. $f(\varnothing) = 0$, to remove any translation effects from our approximation ratios. This is a common maneuver, see [35, 5, 14] for example.

A *polymatroid rank function* $\rho : 2^\Sigma \to \mathbb{Z}$ is a monotone, normalized, submodular function. Though integrality is not strictly required, the polymatroid rank functions used in this work will have integral codomains. A matroid rank function $r_\mathcal{M}$ is simply a *subcardinal* polymatroid rank function, i.e. $r_\mathcal{M}(X) \leq |X|$ always holds. In this way, polymatroids generalize matroids. They can also be viewed as a multiset generalization of matroids as well. These perspectives lead to a few different ways of constructing matroids from polymatroids, which we will see in Section 4.1 and Section 5.2. The *base polytope* of a polymatroid is the set of nonnegative vectors in $\mathbb{R}^\Sigma$ saturating the total rank and satisfying a collection of rank inequalities over all subsets like $\left\{v \in \mathbb{R}^\Sigma_+ \;\middle|\; \|v\|_1 = \rho(\Sigma) \text{ and } (\forall X \subseteq \Sigma)\ \sum_{y \in X} v(y) \leq \rho(X)\right\}$. Famously, when introducing polymatroids Edmonds gave a greedy algorithm for solving linear maximization [11]. For a solution over the base polytope, this is sorting the coordinates by weight and iteratively increasing each coordinate by its marginal return against its predecessors in the sorted order (see [11] for details). Finally, the *span* operator $\sigma_\rho : 2^\Sigma \to 2^\Sigma$ is defined as $\sigma_\rho(X) \triangleq \{y \in \Sigma \mid \rho(X + y) = \rho(X)\}$, i.e. those elements for which the marginal return follows $(\rho/X)(y) = 0$. The span is a closure operator, that is it is *monotone* (i.e. $X \subseteq Y$ implies $\sigma_\rho(X) \subseteq \sigma_\rho(Y)$), *idempotent* (i.e. $\sigma_\rho(\sigma_\rho(X)) = \sigma_\rho(X)$, and *extensive* (i.e. $X \subseteq \sigma_\rho(X)$). We note that the ρ-closed sets are closed under intersection, and that this is a necessary and sufficient description of any closure operator. And so, those sets $X \subseteq \Sigma$

such that $X = \sigma_\rho(X)$ are called ρ-*closed*. Notice, $\sigma_{r_\mathcal{M}} = \sigma_\mathcal{M}$ for a matroid $\mathcal{M}$. A defining feature is the *Steinitz-Maclane exchange property*, which is $y, z \notin \sigma_\mathcal{M}(X)$ and $y \in \sigma_\mathcal{M}(X + z)$ implies $z \in \sigma_\mathcal{M}(X + y)$. This is the reason why we may use matroid span operators to encode substitutability in constraint structures, as the Steinitz-Maclane exchange property ensures that z "capturing" y in the increased span $\sigma_\mathcal{M}(X + z)$ is symmetric. Finally, a *loop* $y \in \Sigma$ is an element such that $\rho(y) = 0$.

Lattices and Order. Let $\mathscr{P} = (\mathcal{X}, \sqsubset)$ be a poset with reflexive closure "$\sqsubseteq$". A set $I \subseteq \mathscr{P}$ is an (order) *ideal* if $\mathbf{z} \in I$ and $\mathbf{y} \sqsubset \mathbf{z}$ implies $\mathbf{y} \in I$. The ideal generated by an arbitrary set S is $\downarrow S \triangleq \{\mathbf{y} \in \mathcal{X} \mid (\exists \mathbf{z} \in S)\ \mathbf{y} \sqsubseteq \mathbf{z}\}$. A *principal ideal* $\downarrow \mathbf{y}$ is generated by a single poset element. For $\mathbf{x}, \mathbf{y} \in \mathscr{P}$ the *meet* $\mathbf{x} \sqcap \mathbf{y}$ is the binary infimum while the *join* $\mathbf{x} \sqcup \mathbf{y}$ is the binary supremum whenever unique such elements exist. A poset is a *lattice* when a meet and join is defined for every pair. The *covering relation*, which we denote by "$\prec$" for any poset, is such that $\mathbf{x} \prec \mathbf{y}$ if and only if $\mathbf{x} \sqsubset \mathbf{y}$ and there is no $\mathbf{z}$ with $\mathbf{x} \sqsubset \mathbf{z} \sqsubset \mathbf{y}$. One visualizes posets via *Hasse diagrams*, consisting of nodes corresponding to the elements of the ground set $\mathcal{X}$ and line segments such that there is a line directed upwards from $\mathbf{x}$ to $\mathbf{y}$ if and only if $\mathbf{x} \prec \mathbf{y}$. A poset $\mathscr{P}$ is *graded* whenever there exists a function $g : \mathscr{P} \to \mathbb{Z}_+$ such that $\mathbf{x} \prec \mathbf{y}$ implies $g(\mathbf{x}) = g(\mathbf{y}) - 1$. A grade taking value zero on bottom elements is the *height function*. Finally, a lattice is *semimodular* if and only if $\mathbf{x} \succ \mathbf{x} \sqcap \mathbf{y}$ implies $\mathbf{y} \prec \mathbf{x} \sqcup \mathbf{y}$. We note semimodular lattices are always graded.

Greedoids. Let Σ^* be all finite sequences over Σ. Each $\alpha \in \Sigma^*$ is a *word* $\alpha = y_1 \dots y_k$ composed of *letters* from Σ. A *language* $\Lambda \subseteq \Sigma^*$ is a set of words. We only examine *simple* languages which do not contain words with repeated letters. To model problems of interest using languages, one can let letters symbolize decisions which can be made over a process, while words give the *feasible* arrangements of decisions. The *support* $\widetilde{\alpha}$ is the set of letters forming α, and we refer to the length of a word via $|\alpha|$. Longest words in Λ are called *basic*. Finally, a *loop* $y \in \Sigma$ is a letter such that $y \notin \widetilde{\alpha}$ for all $\alpha \in \Lambda$, i.e. a loop does not appear in any word. A *greedoid* can be defined as a language generalization of a matroid.

▶ **Definition 1** (Greedoid). *A* greedoid Λ *over* Σ *is a nonempty simple language satisfying,*
1. (Hereditary) $\alpha\beta \in \Lambda$ *implies the prefix* α *is feasible, i.e.* $\alpha \in \Lambda$*,*
2. (Exchange) *For all* $\alpha, \beta \in \Lambda$*, if* $|\alpha| > |\beta|$ *then there exists* $x \in \widetilde{\alpha}$ *such that* $\beta x \in \Lambda$*.*

We give an example relevant to our MPS construction in Section 4, and point the reader towards [25, 2] for more examples. Non-emptiness and the hereditary axiom together imply $\epsilon \in \Lambda$, and exchange makes all basic words equal in length. A prominent class is *interval greedoids*, which possess a stronger variant of exchange: call a simple $\alpha' \in \Sigma^*$ a *subword* of $\alpha \in \Lambda$ if and only if $\widetilde{\alpha'} \subseteq \widetilde{\alpha}$ and the letters of α' occur in an order agreeing with α (i.e. x appears before y in α' implies x appears before y in α).

▶ **Definition 2** (Interval Property). *Let* Λ *be a greedoid.* Λ *has the* interval property *if for all* $\alpha, \beta \in \Lambda$ *with* $|\beta| > |\alpha|$*, there exists a subword* β' *of* β *of length* $|\beta'| \geq |\beta| - |\alpha|$ *with* $\alpha\beta' \in \Lambda$*.*

The contraction minor Λ/α, where $\alpha \in \Lambda$, is the language of simple words which can be concatenated onto α while maintaining feasibility in Λ, i.e. $\Lambda/\alpha \triangleq \{\beta \in \Sigma^* \mid \alpha\beta \in \Lambda\}$. Λ/α is a greedoid, and we call the single length words $y \in \Lambda/\alpha$ *continuations* since they are exactly $y \in \Sigma$ with $\alpha y \in \Lambda$. Define the equivalence relation $\alpha \sim \beta$ by $\Lambda/\alpha = \Lambda/\beta$. A *flat* is an equivalence class of $\Lambda/\sim$, and the class associated with α is $[\alpha]$. One can order the flats by a kind of reachability: let $[\alpha] \sqsubseteq [\beta]$ if and only if there exists $\gamma \in \Lambda/\alpha$ where $\alpha\gamma \in \Lambda$

and $\alpha\gamma \sim \beta$. For interval greedoids, the resulting poset $(\Lambda/\!\sim, \sqsubseteq)$ is a semimodular lattice. Finally, we define the *flat support* as the union of supports of words in a flat. This is given by $\kappa : \Lambda/\!\sim \to 2^\Sigma$ defined as $\kappa[\alpha] \triangleq \bigcup_{\beta\sim\alpha} \widetilde{\beta}$. For interval greedoids κ is invertible, so ordering the flat supports by containment gives a lattice isomorphic to $(\Lambda/\!\sim, \sqsubseteq)$.

A *polymatroid greedoid*, introduced in [24], is such that there exists a polymatroid rank function $\rho : 2^\Sigma \to \mathbb{Z}$ satisfying $\Lambda = \{x_1 \dots x_k \in \Sigma^* \mid (\forall i \in \{1, \dots, k\})\ \rho(\{x_1, \dots, x_i\}) = i\}$. Examples include matroids, poset antimatroids, and certain ordered geometries; see Ch. 7 of [25]. Following [44], in this context we call ρ a *representation* as it encodes how one builds feasible words in Λ. This is, for all $\alpha \in \Lambda$ and $y \notin \widetilde{\alpha}$, $\alpha y \in \Lambda$ if and only if $(\rho/\widetilde{\alpha})(y) = 1$. But more importantly, the representation defines a global submodular structure over the greedoid, which applies to both those sets corresponding to and those which do not correspond to supports of feasible words. This submodular structure is a crucial property given by matroid rank functions which is lost when generalizing from matroids to greedoids. The concept of representation is thus notable because it restores this feature. In [44], the authors define a subclass called *strong polymatroid greedoids* by those guaranteeing the function $\check{\rho}$, where $\check{\rho}(X)$ is the height of a minimal flat support containing X in the lattice $(\kappa(\Lambda/\!\sim), \subseteq)$, i.e. $\check{\rho}(X) \triangleq \min\{|\alpha| \mid X \subseteq \kappa[\alpha]\}$ is a polymatroid rank function representing the greedoid. When true, $\check{\rho}$ is called the *greatest representation* because it can be shown to be coordinatewise larger than all other representations. The authors give a combinatorial description of such greedoids: a greedoid is *optimistic* whenever for every basic word $y_1 \dots y_R$ and nonloop $z \in \Sigma$ there exists an index i making $y_1 \dots y_{i-1} z \in \Lambda$. We remark that all polymatroid greedoids are optimistic [44, 24]. Then, Λ is a strong polymatroid greedoid if and only if it is an optimistic interval greedoid whose flat supports are closed under intersection [44], i.e. $\kappa(F) \cap \kappa(F') = \kappa(F \sqcap F')$ for all $F, F' \in \Lambda/\!\sim$. This shows the existence of a closure operator $\mathrm{cl}(X) \triangleq \bigcap\{\kappa[\alpha] \mid X \subseteq \kappa[\alpha]\}$ which preserves the greedoid structure. Notice, cl also simplifies the definition of $\check{\rho}$ to make $\check{\rho}(X)$ the height of $\mathrm{cl}(X)$ in $(\kappa(\Lambda/\!\sim), \subseteq)$. By inspection then $\sigma_{\check{\rho}} = \mathrm{cl}$, and so the greatest representation gives us functional access to this closure system. For this reason, $\check{\rho}$ and cl are meaningful tools for algorithm design.

4 Matroidal Prerequisite Systems

Now we define a matroidal prerequisite system. This is an interaction between a poset and a matroid, with the poset encoding prerequisites and the matroid encoding a substitutability structure, combined with a compatibility condition (we use this naming to informally refer to Eq. 3) ensuring this interaction is well-behaved.

▶ **Definition 3** (Matroidal Prerequisite System (MPS)). *Fix a finite alphabet Σ, and let $(\Sigma, \mathcal{M})$ and $(\Sigma, \rightsquigarrow)$ be a (loopless) matroid and poset respectively. Define $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ as the set of simple words $x_1 \dots x_\ell \in \Sigma^*$ with $\{x_1, \dots, x_\ell\} \in \mathcal{M}$ and,*

$$y \rightsquigarrow x_i \implies y \in \sigma_{\mathcal{M}}(\{x_1, \dots, x_{i-1}\}) \text{ or } \sigma_{\mathcal{M}}(\{x_1, \dots, x_{i-1}, y\}) = \sigma_{\mathcal{M}}(\{x_1, \dots, x_{i-1}, x_i\}),$$

for all $i \in \{1, \dots, \ell\}$. Then, $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is a matroidal prerequisite system *if and only if,*

$$\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow) \text{ and } z \in \sigma_{\mathcal{M}}(\widetilde{\alpha}) \implies (\forall y \rightsquigarrow z)\ y \in \sigma_{\mathcal{M}}(\widetilde{\alpha}). \tag{3}$$

This is, $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is words with independent supports such that every prerequisite of a letter y in a feasible word is either spanned by prefix ending at y (exclusive) in the matroid or is substitutable with y against that prefix. Call the words of $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ *feasible*. Then, the compatibility condition in Eq. 3 states that whenever a letter is covered by the matroid span

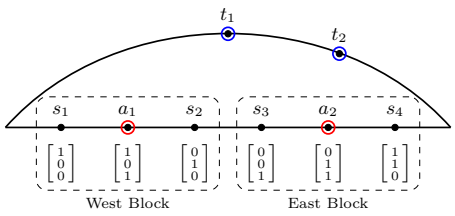
of a feasible word, so must all of its prerequisites. This has good semantic meaning with the problems we are interested in modeling (i.e. a decision which has been made redundant is such that all of its prerequisite decisions are redundant as well), and is required to verify $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is an optimistic greedoid (which we need to use $\check{\rho}$, cl, and other concepts inducing correct behavior in our algorithms). We also implicitly assume the matroid is loopless so as to prevent complications or cumbersome notation caused by loops. This can be satisfied by simply removing letters with matroid rank zero from the groundset. Finally, as discussed in Section 1, this abstracts matroids and poset antimatroids by considering the MPS defined over an empty order and free matroid, respectively.

Before continuing, we list a few properties. Firstly, every basic word of $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is always supported on a basic set (largest independent set) in $\mathcal{M}$. This is because, for all $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$, any minimal (w.r.t. "$\rightsquigarrow$") letter $y \in \Sigma \setminus \sigma_{\mathcal{M}}(\widetilde{\alpha})$ is such that minimality guarantees all its prerequisites are covered by $\sigma_{\mathcal{M}}(\widetilde{\alpha})$. Moreover, $y \notin \sigma_{\mathcal{M}}(\widetilde{\alpha})$ also implies $\widetilde{\alpha} + y$ is independent, and so $\alpha y \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$. Such a y always exists whenever $\sigma_{\mathcal{M}}(\widetilde{\alpha}) \neq \Sigma$, which occurs exactly when $\widetilde{\alpha}$ is not a basic set of the matroid. Secondly, this implies every MPS defined by a loopless matroid is also loopless. Specifically, since $\sigma_{\mathcal{M}}(\widetilde{\alpha}) = \Sigma$ for every basic word α, for all $y \in \Sigma$ there exists an α-prefix $\alpha' z$ such that $y \in \sigma_{\mathcal{M}}(\widetilde{\alpha'} + z) \setminus \sigma_{\mathcal{M}}(\widetilde{\alpha'})$. Then, since y is $\widetilde{\alpha'}$-substitutable with all its prerequisites by the compatibility condition (Eq. 3), it follows that $\alpha' y \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$. In what comes, we show an MPS is optimistic, and so this looplessness will mean that given any basic word and letter $y \in \Sigma$ we can assert the existence of a prefix possessing y as a continuation.

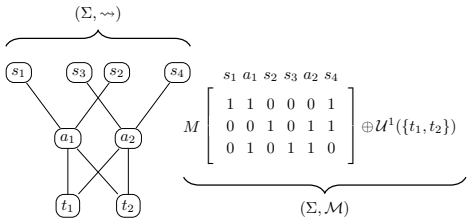
As a technical example, let $\mathcal{U}_n^R$ be a *uniform matroid* of rank R. That is, every set of cardinality at most R is independent. $\mathcal{L}(\mathcal{U}_n^R, \rightsquigarrow)$ always satisfies the compatibility condition, regardless of the prerequisite order $(\Sigma, \rightsquigarrow)$. This is because the only independent sets with span not equal to themselves are basic. So, basic words in $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ are the only feasible words whose corresponding matroid span contains any letters not from their support. Therefore, for all $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ which are not basic we have $z \in \sigma_{\mathcal{M}}(\widetilde{\alpha})$ implies $z \in \widetilde{\alpha}$, and so all prerequisites $y \rightsquigarrow z$ are contained in $\sigma_{\mathcal{M}}(\widetilde{\alpha})$ by MPS feasbility, while if α is basic then it is supported on a basic set of $\mathcal{U}_n^R$ and $\sigma_{\mathcal{M}}(\widetilde{\alpha}) = \Sigma$ trivially makes the conclusion of Eq. 3 true. This makes $\mathcal{L}(\mathcal{U}_n^R, \rightsquigarrow)$ a matroidal prerequisite system. We can use this to model cardinality constrained optimization over a poset antimatroid. This is equivalent to adding a cardinality constraint on the previously studied maximum closure over a directed acyclic graph [38], and restricting to unit-weights for previously studied poset constrained knapsacks [19, 41, 20].

▶ **Example 4** (Cardinality Constrained Optimization on a Poset Antimatroid)**.** Let $(\Sigma, \rightsquigarrow)$ be a poset with $|\Sigma| = n$, $w : \Sigma \to \mathbb{R}$ a nonnegative weight function, and $k \leq n$ an integer. Our task is to find an ideal of size k of maximum cumulative weight, or equivalently finding a k-length prefix of a linear extension with maximum cumulative weight. Add an additional "sentinel letter" t by letting $\Sigma' = \Sigma + t$, and consider the matroid $(\Sigma', \mathcal{U}_{n+1}^{k+1})$. Extend the order to $(\Sigma', \rightsquigarrow')$ by letting $y \rightsquigarrow z \implies y \rightsquigarrow' z$ and $y \rightsquigarrow' t$ for all $y \in \Sigma$. This is, the derived order inherits the relations given by "$\rightsquigarrow$" while making every letter lesser than the sentinel. Finally, make the derived weight function $w' : \Sigma' \to \mathbb{R}$ by letting $w'(y) = w(y)$ for all $y \in \Sigma$, and $w'(t) = \max_{y \in \Sigma} w(y) + 1$. It can be seen that any word of maximum weight over the MPS $\mathcal{L}(\mathcal{U}_{n+1}^{k+1}, \rightsquigarrow')$ is a basic word ending in t. It follows that the first k letters of such a word will be an ideal of $(\Sigma, \rightsquigarrow)$ of size k maximizing $X \mapsto \sum_{y \in X} w(y)$.

As a less technical example, we discuss how our MPS can be used to enrich the constraint structure in sensor placement problems.



(a) A bridge with sensor placements. Modal signatures are underneath each sensor. Baseline sensors are outlined in red, sync hardware is outlined in blue, and the remaining sensors are for anomaly detection.



(b) A MPS following Example 5 is given by a poset like above and the direct sum of a linear matroid on GF(2) over the modal signatures with the rank 1 uniform matroid over $\{t_1, t_2\}$.

Figure 1 Following Example 5, modal signatures of the baseline sensor of each block must be spanned before an anomaly detection sensor is placed in that block. Furthermore, we must place sync hardware before any other sensor. This leads to the poset structure on the right. For budget reasons, we can only place one sync hardware. A linear matroid encodes the substitution structure of the signatures as well as a linear independence condition in the measurements, so the matroid is given by a direct sum like on the right. To save space, verifying the compatibility condition is deferred to the full version. Some examples of basic words are $t_1a_1s_1s_2$, $t_1a_2s_4s_1$, and $t_2a_1a_2s_1$.

▶ **Example 5** (Sensor Networks for Anomaly Detection)**.** Sensor placement to maximize mutual information [26] is a classic instance of monotone submodular maximization, with particularly natural matroid constraints encoding budget, network interference, etc. (see [39, 8] for example). One task is using sensors to detect anomalies; for example the presence of errant vibration frequencies indicates damage in a structure [1]. However, it is not possible to interpret anomalies without a frame of reference given by baseline measurements. This creates a prerequisite structure, where modes corresponding to a baseline must be spanned before anomaly detection can occur. Other operational prerequisites may coexist, like requiring synchronization hardware is placed before other hardware to ensure a correct boot procedure.

An instance of detecting anomalous frequencies on a bridge is shown in Figure 1. A modal signature is a binary vector encoding the capabilities of each sensor, and a linear matroid in combination with a poset encodes the requirement that baseline sensor modal signatures in each block must be spanned before anomaly detecting sensors in those blocks can be placed, along with the requirement that clock synchronization hardware is placed before any sensor. An interesting feasible word is $t_1a_2s_4s_1$. Even though the baseline sensor a_1 was never placed in the East block, we still place s_1 after $t_1a_2s_4$ since the modal signature of a_1 is in the span of $\{a_2, s_4\}$. This is because the functionality of a_1 is redundant in the presence of a_2 and s_4.

In Section 4.1, we examine the greedoid theory of an MPS. In particular, we show every MPS can be described as a strong polymatroid greedoid and vice versa. The utility of this is best seen in Section 4.3, where we use the greatest representation and closure operator of strong polymatroid greedoids to give procedures for "filling-in" and "projecting" partial optimization solutions on to the basic words of an MPS. This is Lemma 11, which will be the foundation of our algorithms in Sections 5 and 6. Finally, since we use the greatest representation in our algorithms, we verify that it can be computed efficiently when given a matroid rank oracle in Section 4.2. This is because, even though our algorithms are described in terms of strong polymatroid greedoid concepts, our previous examples show that the MPS is a widely applicable and interpretable constraint model. And so, while strong polymatroid greedoids give the mechanisms for our algorithms, an MPS grounds our motivations for designing them in the first place. Therefore, we must verify that given an MPS it is possible to efficiently compute strong polymatroid greedoid concepts under reasonable assumptions.

4.1 Greedoid Theoretic Aspects

We now develop the greedoid theory of an MPS. In particular, we will show that a greedoid is a strong polymatroid greedoid if and only if there exists a matroid and poset giving its feasible words as an MPS. We first verify an MPS is an optimistic interval greedoid.

▶ **Proposition 6.** *Let $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ be a matroidal prerequisite system. Then, $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is an optimistic interval greedoid.*

Proof. From the definition $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is nonempty and hereditary. Let $\alpha, \beta \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ with $|\beta| > |\alpha|$. Select the first z_1 appearing in $\widetilde{\beta}$ with $z_1 \notin \sigma_{\mathcal{M}}(\widetilde{\alpha})$. We show αz_1 is feasible. Well, $z_1 \notin \sigma_{\mathcal{M}}(\widetilde{\alpha})$ implies $\widetilde{\alpha} + z_1$ is independent. And, if $\beta' z_1$ is the β-prefix ending at z_1, we see $\widetilde{\beta'} \subseteq \sigma_{\mathcal{M}}(\widetilde{\alpha})$ by construction, thus $\sigma_{\mathcal{M}}(\widetilde{\beta'}) \subseteq \sigma_{\mathcal{M}}(\widetilde{\alpha})$. So, for every $y \rightsquigarrow z_1$ such that $y \notin \sigma_{\mathcal{M}}(\widetilde{\alpha})$, it follows that $y \notin \sigma_{\mathcal{M}}(\widetilde{\beta'})$. Since $\beta' \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$, as we have already shown the hereditary property, it must be that y and z_1 are $\widetilde{\beta'}$-substitutable. Hence,

$$\sigma_{\mathcal{M}}(\widetilde{\alpha} + y) = \sigma_{\mathcal{M}}(\widetilde{\alpha} \cup \sigma_{\mathcal{M}}(\widetilde{\beta'} + y)) = \sigma_{\mathcal{M}}(\widetilde{\alpha} \cup \sigma_{\mathcal{M}}(\widetilde{\beta'} + z_1) = \sigma_{\mathcal{M}}(\widetilde{\alpha} + z_1),$$

by $\sigma_{\mathcal{M}}(\widetilde{\beta'}) \subseteq \sigma_{\mathcal{M}}(\widetilde{\alpha})$, and so y and z_1 are $\widetilde{\alpha}$-substitutable. Thus $\alpha z_1 \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$. If $|\beta| - |\alpha| > 1$, one can repeat this argument on the first letter z_2 in β with $z_2 \notin \sigma_{\mathcal{M}}(\widetilde{\alpha z_1})$ to show $\alpha z_1 z_2 \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ as well. Repeating in this way $|\beta| - |\alpha| - 2$ more times shows the existence of the $k = |\beta| - |\alpha|$ length β-subword $z_1 \dots z_k$ such that $\alpha z_1 \dots z_k \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$. And so, $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is an interval greedoid.

Now, let $x_1 \dots x_R \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ be basic. Select $y \in \Sigma$ and let i be the smallest index with $y \in \sigma_{\mathcal{M}}(\{x_1, \dots, x_i\})$. By Steinitz-Maclane exchange property, x_i and y are $\{x_1, \dots, x_{i-1}\}$-substitutes. Well, for every $z \rightsquigarrow y$ with $z \notin \sigma_{\mathcal{M}}(\{x_1, \dots, x_{i-1}\})$, the compatibility condition and the observation that x_i and y are $\{x_1, \dots, x_{i-1}\}$-substitutes combine for,

$$y \in \sigma_{\mathcal{M}}(\{x_1, \dots, x_{i-1}, x_i\}) \implies z \in \sigma_{\mathcal{M}}(\{x_1, \dots, x_{i-1}, y\}).$$

So, $x_1 \dots x_{i-1} y \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$, and optimism follows. ◀

Now, we make observations about the interaction between the flat supports, matroid, and prerequisite order. Specifically, the next lemma first verifies flat supports are always $r_{\mathcal{M}}$-closed and ideals of $(\Sigma, \rightsquigarrow)$. This then shows the flat supports are closed under intersection.

▶ **Lemma 7.** *Let $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ be a matroidal prerequisite system. The following are all true:*
1. *For all $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$, $\sigma_{\mathcal{M}}(\widetilde{\alpha}) = \kappa[\alpha]$,*
2. *Every flat support is an ideal in the prerequisite order $(\Sigma, \rightsquigarrow)$,*
3. *The flat supports are closed under intersection.*

To prove the first item, we need the following statement which follows from the semimodularity of the flats of interval greedoids. See [2] for proof.

▶ **Lemma 8** ([2])**.** *Let Λ be an interval greedoid, and select nonempty words $\alpha x \in \Lambda$ and $\alpha y \in \Lambda$. If $\alpha x \not\sim \alpha y$, then $\alpha x y \in \Lambda$ and $\alpha y x \in \Lambda$.*

Proof of Lemma 7. For what follows, recall Proposition 6 shows $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is an optimistic interval greedoid. Let $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ and select $y \in \sigma_{\mathcal{M}}(\widetilde{\alpha})$. Since supports of feasible words are independent, there exists no word in the contraction $\mathcal{L}(\mathcal{M}, \rightsquigarrow)/\alpha$ containing y in its support. So by optimism (since every feasible word can be extended into a basic word), there exists an α-prefix $\alpha' z$ with $y \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)/\alpha'$. Let $\alpha' z$ be a longest such prefix, so that $\alpha' y, \alpha' z \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ while neither $\alpha' y z$ or $\alpha' z y$ are feasible. By contrapositive of Lemma 8, it

follows that $\alpha' y \sim \alpha' z$. Because our construction makes $[\alpha' z] \sqsubseteq [\alpha]$, it follows that $[\alpha' y] \sqsubseteq [\alpha]$. So, there exists $\beta \in \Sigma^*$ making $\alpha' y \beta \in [\alpha]$, thus $y \in \kappa[\alpha]$. This shows $\sigma_{\mathcal{M}}(\widetilde{\alpha}) \subseteq \kappa[\alpha]$. Now, select $z' \notin \sigma_{\mathcal{M}}(\widetilde{\alpha})$. Then, there exists $\beta \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)/\alpha$ with $z' \in \widetilde{\beta}$: There must exist a minimal $y' \in {\downarrow} z' \setminus \sigma_{\mathcal{M}}(\widetilde{\alpha})$ w.r.t. "$\rightsquigarrow$", and this minimal y' must be a continuation of α since its prerequisites are covered by $\sigma_{\mathcal{M}}(\widetilde{\alpha})$. Therefore, if $\alpha z' \notin \mathcal{L}(\mathcal{M}, \rightsquigarrow)$, we can append y' onto α. If we continue this process, eventually z' will be substitutable with every prerequisite that hasn't been covered, at which point z' will be a continuation. This makes $z' \notin \kappa[\alpha]$. So, as we've shown $\sigma_{\mathcal{M}}[\alpha] \subseteq \kappa[\alpha]$ and $\Sigma \setminus \sigma_{\mathcal{M}}(\widetilde{\alpha}) \subseteq \Sigma \setminus \kappa[\alpha]$, it follows that $\sigma_{\mathcal{M}}(\widetilde{\alpha}) = \kappa[\alpha]$. This gives Lemma 7.1. Finally, we note that our compatibility condition in Eq. 3 makes $\sigma_{\mathcal{M}}(\widetilde{\alpha})$ equal to an ideal in $(\Sigma, \rightsquigarrow)$. Since we've shown $\kappa[\alpha] = \sigma_{\mathcal{M}}(\widetilde{\alpha})$, this gives Lemma 7.2.

This description of the flat supports is convenient for showing they are closed under intersection. In particular, since ideals of a poset are always closed under intersection, it follows that $\kappa(F) \cap \kappa(F')$ is an ideal on $(\Sigma, \rightsquigarrow)$ for all $F, F' \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)/\sim$. It is also true that $\kappa(F \sqcap F')$ and $\kappa(F) \cap \kappa(F')$ are closed in $\mathcal{M}$. Now, observe that the inclusion $\kappa(F \sqcap F') \subseteq \kappa(F) \cap \kappa(F')$ is always true by $(\Lambda/\sim, \sqsubseteq) \cong (\kappa(\Lambda/\sim), \subseteq)$, so for 2. to hold we need only verify the reverse inclusion. By way of contradiction, suppose there exists $z \in (\kappa(F) \cap \kappa(F')) \setminus \kappa(F \sqcap F')$. Without loss of generality, let z be a minimal letter in this difference w.r.t. "$\rightsquigarrow$". Because $\kappa(F \sqcap F')$ is an ideal, minimality implies $y \in \kappa(F \sqcap F')$ for all $y \rightsquigarrow z$. Therefore, for all $\mu \in F \sqcap F'$, every prerequisite of z is covered by the span $\sigma_{\mathcal{M}}(\widetilde{\mu})$ since $\sigma_{\mathcal{M}}(\widetilde{\mu}) = \kappa(F \sqcap F')$ by 1. This makes z a continuation of words in $F \sqcap F'$, hence $F \sqcap F' \prec [\mu z]$. But, $z \in \kappa(F) \cap \kappa(F')$ implies $[\mu z]$ is lesser than both F and F'. One (of many) ways to see why is because if we select a word $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)/\mu$ making $\mu\alpha \in F$, then by exchange there is an α-subword α' of length $|\alpha| - 1$ such that $\mu z \alpha'$ is feasible. This construction makes both $\widetilde{\mu z \alpha'} \subseteq \kappa(F)$, and the flats F and $[\mu z \alpha']$ the same height in $(\Lambda/\sim, \sqsubseteq)$. So, by $(\Lambda/\sim, \sqsubseteq) \cong (\kappa(\Lambda/\sim), \subseteq)$, this makes $\mu z \alpha' \in F$. A symmetric argument applies to show $[\mu z] \sqsubseteq F'$ as well. Therefore $[\mu z] \sqsubseteq F \sqcap F'$. Yet, this contradicts $F \sqcap F' \prec [\mu z]$, so $\kappa(F) \cap \kappa(F') = \kappa(F \sqcap F')$. So we've shown Lemma 7.3. ◀

Therefore, by Lemma 7.3 and Proposition 6, every MPS is a strong polymatroid greedoid by the description from [44]. For logical completeness, we give a construction encoding every strong polymatroid greedoid as an MPS. Because the forthcoming results only use the direction already proven, we defer this to the full version to satisfy space constraints.

▶ **Theorem 9.** *Let Λ be a greedoid on the alphabet Σ. There exists a matroid $(\Sigma, \mathcal{M})$ and compatible poset $(\Sigma, \rightsquigarrow)$ such that Λ equals the words of the matroidal prerequisite system $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ if and only if Λ is a strong polymatroid greedoid.*

4.2 Computing the Greatest Representation

Our algorithms use the greatest representation $\check{\rho}$. To ensure this is well-founded, we show $\check{\rho}$ can be evaluated via a single rank query on the underlying matroid. Specifically, the greatest representation of an MPS $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is the composition of the matroid rank and ideal operator on $(\Sigma, \rightsquigarrow)$. Note that ideals can be computed efficiently in $\mathsf{poly}(|\Sigma|)$-time by repeated ordered comparisons.

▶ **Proposition 10.** *Let $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ be an MPS. Then, $\check{\rho} = r_{\mathcal{M}} \circ {\downarrow}$.*

Proof. Fix $X \subseteq \Sigma$. We verify $\check{\rho}(X) = r_{\mathcal{M}}({\downarrow} X)$. Perform the following procedure: Begin with a word $\alpha = \epsilon$. Then, in each step, select a smallest $z \in {\downarrow} X \setminus \widetilde{\alpha}$ satisfying $r_{\mathcal{M}}(\widetilde{\alpha} + z) > r_{\mathcal{M}}(\widetilde{\alpha})$. Terminate once no such z exists. See that $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ because, fixing z to a selection made in any round, minimality makes every $y \rightsquigarrow z$ spanned by the round's value of α. Moreover,

it is the case that $(r_{\mathcal{M}}/\widetilde{\alpha})(\downarrow X) = 0$ upon termination. So, with α being the value at the end of the final round, we have $\downarrow X \subseteq \sigma_{\mathcal{M}}(\widetilde{\alpha})$. Yet combining this with $\widetilde{\alpha} \subseteq \downarrow X$ makes $r_{\mathcal{M}}(\widetilde{\alpha}) \leq r_{\mathcal{M}}(\downarrow X) \leq r_{\mathcal{M}}(\widetilde{\alpha})$, so $r_{\mathcal{M}}(\downarrow X) = |\alpha|$ since $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ implies $\widetilde{\alpha} \in \mathcal{M}$. Now, take note that $\downarrow X \subseteq \sigma_{\mathcal{M}}(\widetilde{\alpha})$ also implies that no $\beta \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)/\alpha$ satisfies $\widetilde{\beta} \cap \downarrow X \neq \varnothing$, as the supports of feasible words must be independent. This makes $(\check{\rho}/\widetilde{\alpha})(y) \neq 1$ for all $y \in \downarrow X$ in particular. By optimism (see Proposition 6) it follows that there exists an α-prefix $\alpha^{(y)}z$ for each $y \in \downarrow X$ such that $\alpha^{(y)}y$ is feasible. So, $(\check{\rho}/\widetilde{\alpha}^{(y)})(y) = 1$. But by the law of diminishing returns and $(\check{\rho}/\widetilde{\alpha})(y) \neq 1$ it follows that $1 = (\check{\rho}/\widetilde{\alpha}^{(y)})(y) > (\check{\rho}/\widetilde{\alpha})(y)$ for all $y \in \downarrow X$. And so, as $\check{\rho}$ is integral, we have $(\check{\rho}/\widetilde{\alpha})(y) = 0$ for all $y \in \downarrow X$. This makes $\downarrow X \subseteq \mathrm{cl}(\widetilde{\alpha})$. Therefore $\check{\rho}(\widetilde{\alpha}) \leq \check{\rho}(\downarrow X) \leq \check{\rho}(\widetilde{\alpha})$, and so $\check{\rho}(\downarrow X) = \check{\rho}(\widetilde{\alpha})$. But, because flat supports are ideals by Lemma 7.2, we see $\downarrow X \subseteq \mathrm{cl}(X)$, and so $\check{\rho}(X) \leq \check{\rho}(\downarrow X) \leq \check{\rho}(X)$ by $\mathrm{cl} = \sigma_{\check{\rho}}$. Therefore, $\check{\rho}(X) = \check{\rho}(\widetilde{\alpha})$, and since $\check{\rho}$ is a representation we have $\check{\rho}(\widetilde{\alpha}) = |\alpha|$. Hence we've found $r_{\mathcal{M}}(\downarrow X) = |\alpha|$ and $\check{\rho}(X) = |\alpha|$. ◀

4.3 Projection and Fill-In

In Sections 5 we present algorithms based on projecting a relaxed solution back onto the words of an MPS, while in Section 6 our algorithms build words through repeated insertions eventually converging to a feasible word. Both ideas require a description of what sets of letters, which might not support a feasible word, can be "filled" into a feasible word. In the following, we show that words $x_1 \dots x_\ell$ such that $(\check{\rho}/\{x_1, \dots, x_{i-1}\})(x_i) > 0$ for all $i \in \{1, \dots, \ell\}$ satisfy this. We note this is a special property of strong polymatroid greedoids.

▶ **Lemma 11.** *Suppose Λ is a strong polymatroid greedoid. Then, for all simple $x_1 \dots x_k \in \Sigma^*$, if $(\check{\rho}/\{x_1, \dots, x_{i-1}\})(x_i) > 0$ for all $i \in \{1, \dots, k\}$, there exists a set of letters $\{y_{i,1}, \dots, y_{i,\ell_i}\}$ with $\ell_i = (\check{\rho}/\{x_1, \dots, x_{i-1}\})(x_i) - 1$ inducing the feasible word,*

$$y_{1,1} \dots y_{1,\ell_1} x_1 y_{2,1} \dots y_{2,\ell_2} x_2 \dots y_{k,1} \dots y_{k,\ell_k} x_k \in \Lambda.$$

Proof. By $(\check{\rho}/\{x_1, \dots, x_{i-1}\})(x_i) > 0$ for all $i \in \{1, \dots, k\}$, the existence of the strictly ascending chain $\varnothing \subsetneq \mathrm{cl}(\{x_1\}) \subsetneq \mathrm{cl}(\{x_1, x_2\}) \subsetneq \dots \subsetneq \mathrm{cl}(\{x_1, \dots, x_k\})$ of flat supports in $(\kappa(\Lambda/{\sim}), \subseteq)$ follows. Then, select letters $\{y_{i,1}, \dots, y_{i,\ell_i}\}$ for each x_i inducing a covering chain beginning at $\mathrm{cl}(\{x_1, \dots, x_{i-1}\})$ and ending at $\mathrm{cl}(\{x_1, \dots, x_i\})$ like,

$$\begin{aligned} \mathrm{cl}(\{x_1, \dots, x_{i-1}\}) \prec \mathrm{cl}(\{x_1, \dots, x_{i-1}, y_{i,1}\}) \prec \dots \prec \mathrm{cl}(\{x_1, \dots, x_{i-1}, y_{i,1}, \dots, y_{i,\ell_i}\}), \\ \prec \mathrm{cl}(\{x_1, \dots, x_{i-1}, y_{i,1}, \dots, y_{i,\ell_i}, x_i\}) = \mathrm{cl}(\{x_1, \dots, x_i\}). \end{aligned} \tag{4}$$

This is always possible because cl is a closure operator. In particular, since $\sigma_{\check{\rho}} = \mathrm{cl}$, having already selected $y_{i,1}, \dots, y_{i,j-1}$, one takes an arbitrary $y_{i,j}$ satisfying both of,

$$\begin{cases} (\check{\rho}/\{x_1, \dots, x_{i-1}, y_{i,1}, \dots, y_{i,j-1}\})(y_{i,j}) = 1, \\ \mathrm{cl}(\{x_1, \dots, x_{i-1}, y_{i,1}, \dots, y_{i,j-1}\} + y_{i,j}) \subseteq \mathrm{cl}(\{x_1, \dots, x_i\}), \end{cases} \tag{5}$$

whenever $(\check{\rho}/\{x_1, \dots, x_{i-1}, y_{i,1}, \dots, y_{i,j-1}\})(x_i) \neq 1$, and terminates otherwise. A letter $y_{i,j}$ satisfying the conditions in Eq. 5 must exist when $(\check{\rho}/\{x_1, \dots, x_{i-1}, y_{i,1}, \dots, y_{i,j-1}\})(x_i) \neq 1$, as there is a flat support below $\mathrm{cl}(\{x_1, \dots, x_i\})$ covering $\mathrm{cl}(\{x_1, \dots, x_{i-1}, y_{i,1}, \dots, y_{i,j-1}\})$ in $(\kappa(\Lambda/{\sim}), \subseteq)$, and any letter in the difference of this cover and $\mathrm{cl}(\{x_1, \dots, x_{i-1}, y_{i,1}, \dots, y_{i,j-1}\})$ will satisfy the conditions in Eq. 5. Now, recall that $\check{\rho}(X)$ is defined as the height of the least flat support containing X (recall flat supports are closed under intersection, so a least such flat is unique). So, $\check{\rho}$ grades the cl-image. Then, the covering chain structure from Eq. 4 and $\sigma_{\check{\rho}} = \mathrm{cl}$ makes,

$$(\overset{\vee}{\rho}/\{x_1,\ldots,x_{i-1},y_{i,1},\ldots,y_{i,j-1}\})(y_{i,j}) = 1, \quad \forall j \in \{1,\ldots,\ell_i\}. \tag{6}$$

This means, for example, that $y_{1,1}\ldots y_{1,\ell_1}x_1 \in \Lambda$ by the definition of representation in polymatroid greedoids. Then, since selecting like Eq. 4 makes $\mathrm{cl}(\{y_{1,1},\ldots,y_{1,\ell_1},x_1\}) = \mathrm{cl}(\{x_1\})$, applying the same argument using Eq. 6 for $i = 2$ shows $y_{1,1}\ldots y_{1,\ell_1}x_1y_{2,1}\ldots y_{2,\ell_2}x_2 \in \Lambda$. One repeats this procedure up to $i = k$ to get the result. ◀

5 Scaffolding Solutions via Polymatroid Relaxations

Observe, every polymatroid greedoid is such that the characteristic vectors of its basic words are base vectors of any of its polymatroid representations. To see why, note that Edmonds' algorithm [11] over the base polytope always returns an extreme point. Let $y_1\ldots y_R \in \mathcal{L}(\mathcal{M},\rightsquigarrow)$ be any basic word of an MPS with greatest representation $\overset{\vee}{\rho}$. Then, a weight function $w : \Sigma \to \mathbb{R}$ such that $w(y_i) = R - (i-1)$ for all $i \in \{1,\ldots,R\}$ and $w(z) = 0$ for $z \notin \{y_1,\ldots y_R\}$ will make Edmonds' algorithm examine coordinates in the order dictated by the basic word. Because $\overset{\vee}{\rho}$ is a representation, we see that $(\overset{\vee}{\rho}/\{y_1,\ldots,y_{i-1}\})(y_i) = 1$ for all i, and Lemma 11 certifies that this ends at y_R, as the existence of $z \in \Sigma$ with $(\overset{\vee}{\rho}/\{y_1,\ldots,y_R\})(z) > 0$ would imply $y_1\ldots y_R$ is not basic. Therefore, Edmonds' algorithm will increase each coordinate from $\{y_1,\ldots,y_R\}$ to exactly one and terminate on the characteristic vector of the support of the basic word.

In what follows we will examine nonnegative additive maximization and monotone submodular maximization algorithms based on using the greatest representation as a relaxation. Specifically, Section 5.1 will first describe a method for additive maximization in Algorithm 1 which is based on computing a relaxed solution via Edmonds' algorithm and projecting onto a basic word using Lemma 11. In Section 5.2 we approach submodular maximization using the continuous greedy algorithm [5] to compute a set of weights forming an additive underestimation of the objective, which we give to our Algorithm 1 to compute a solution on the MPS. The resulting guarantees will vary in largest matroid rank of a principal ideal Δ.

5.1 Additive Maximization

Let $w : \Sigma \to \mathbb{R}_+$ be a nonnegative weight function. Our task is to find a feasible word α of a MPS $\mathcal{L}(\mathcal{M},\rightsquigarrow)$ maximizing $\alpha \mapsto \sum_{y\in\tilde{\alpha}} w(y)$. Let $v^\star \in \mathbb{R}^\Sigma$ be base vector of $\overset{\vee}{\rho}$ computed by Edmonds' algorithm, and sort the nonzero coordinates $\{x_1,x_2,\ldots\} = \mathrm{supp}(v^*)$ so that $w(x_1) \geq w(x_2) \geq \ldots$. This is, we sort the support of $v^\star$ in order of the coordinate increases made by Edmonds' algorithm. Since $x_i \in \mathrm{supp}(v^\star)$, it follows that $(\overset{\vee}{\rho}/\{x_1,\ldots,x_{i-1}\})(x_i) > 0$ for all $i \in \{1,\ldots,|\mathrm{supp}(v^\star)|\}$. This satisfies the premise of Lemma 11, and so perform a fill-in procedure like in the proof of Lemma 11 to obtain a feasible word containing $\mathrm{supp}(v^\star)$.

This idea is shown in Algorithm 1. We first compute a "scaffold" $v^\star$ for the solution using Edmonds' greedy algorithm on $\overset{\vee}{\rho}$ in line 1, whose support we then sort in line 2. Then, we use the same arbitrary fill-in scheme presented in the proof of Lemma 11, in particular Eq. 5, to obtain a feasible word α such that $\mathrm{supp}(v^\star) \subseteq \widetilde{\alpha}$ in lines 4 through 8. Note that the while loop cannot terminate with $(\overset{\vee}{\rho}/\widetilde{\alpha})(x_i) = 0$: if a selected fill-in letter y with $(\overset{\vee}{\rho}/\widetilde{\alpha})(y) = 1$ made $x_i \in \mathrm{cl}(\widetilde{\alpha}+y)$, then we would have $(\overset{\vee}{\rho}/\widetilde{\alpha})(x_i) \leq 1$ which contradicts the loop condition.

Recall, in Section 1 we defined Δ as the maximum matroid rank of a principal ideal of $(\Sigma,\rightsquigarrow)$. By the law of diminishing returns, it follows that each coordinate of $v^\star$ is at most Δ. When comparing $v^\star$ and $\chi_{\tilde{\alpha}}$, it follows that any entry in $v^\star$ is at most a Δ-factor smaller in $\chi_{\tilde{\alpha}}$. This idea shows Algorithm 1 computes a Δ-approximation.

Algorithm 1 Scaffolded Greedy with Arbitrary Fill-In.

1 $v^\star \leftarrow$ solution from Edmonds' algorithm [11] on $\check{\rho}$ with weights w
2 select $x_1 \dots x_k \in \mathrm{supp}(v^\star)!$ with $w(x_1) \geq w(x_2) \geq \dots \geq w(x_k)$
3 $\alpha \leftarrow \epsilon$
4 **for** $i \in \{1, \dots, k\}$ **do**
5 **while** $(\check{\rho}/\widetilde{\alpha})(x_i) > 1$ **do**
6 select $y \in \Sigma$ with $\big(\check{\rho}/\widetilde{\alpha}\big)(y) = 1$ and $\widetilde{\alpha} + y \subseteq \mathrm{cl}(\{x_1, \dots, x_i\})$
7 $\alpha \leftarrow \alpha y$
8 $\alpha \leftarrow \alpha x_i$
9 **return** α

▶ **Theorem 12.** *Let $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ be a matroidal prerequisite system, $w : \Sigma \to \mathbb{R}_+$ a nonnegative weight function, and* $\mathrm{OPT} = \max_{\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)} \sum_{y \in \widetilde{\alpha}} w(y)$. *Algorithm 1 outputs $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ with $\sum_{y \in \tilde{\alpha}} w(y) \geq \Delta^{-1} \cdot \mathrm{OPT}$.*

Proof. Firstly, one sees that the fill-in (lines 4-8) is done in the same way as the proof of Lemma 11, so it is clear from our discussion at the start of the section that the resulting α is feasible. Moreover, it is basic as $\check{\rho}(\mathrm{supp}(v^\star)) = \check{\rho}(\Sigma)$ and $\mathrm{cl}(\widetilde{\alpha}) = \mathrm{cl}(\mathrm{supp}(v^\star))$ makes $\check{\rho}(\widetilde{\alpha}) = \check{\rho}(\Sigma)$. For each $x_i \in \mathrm{supp}(v^\star)$, by definition of Edmonds' algorithm (see [11]) we have $v^\star(x_i) = (\check{\rho}/\{x_1, \dots, x_{i-1}\})(x_i)$. Therefore, for any choice of $y \in \mathrm{supp}(v^\star)$, we have $v^\star(y) \leq \check{\rho}(y)$ by the law of diminishing returns. Of course $\check{\rho}(y) \leq \Delta$, and so,

$$\chi_{\tilde{\alpha}}(y) \geq \frac{1}{\Delta} \cdot v^\star(y), \quad \forall y \in \mathrm{supp}(v^\star). \tag{7}$$

Then, let α_{OPT} be an optimal word. Recall by our discussion at the beginning of Section 5 that $\chi_{\widetilde{\alpha}_{\mathrm{OPT}}}$ is an extreme point of the base polytope, and so as Edmonds' algorithm is optimal over this constraint set we see $\mathrm{OPT} = w^\intercal \chi_{\tilde{\alpha}_{\mathrm{OPT}}} \leq w^\intercal v^\star$. Combining this with Eq. 7 shows the desired $w^\intercal \chi_{\tilde{\alpha}} \geq \Delta^{-1} \cdot w^\intercal v^\star \geq \Delta^{-1} \cdot \mathrm{OPT}$. ◀

5.2 Monotone Submodular Maximization

For what comes, we require the following: Every integral polymatroid can be associated with a matroid via the *natural matroid* construction [16]. Specifically, define the ground set as,

$$E_{\check{\rho}} \triangleq \bigcup_{y \in \Sigma} \{(y, 1), \dots, (y, \check{\rho}(y))\}.$$

This just introduces $\check{\rho}(y)$ copies of each letter $y \in \Sigma$. This is equivalently seen as a multiset, so let the multiplicity vector of $S \subseteq E_{\check{\rho}}$ be given by $\mu_S(y) \triangleq |S \cap (\{y\} \times \{1, \dots, \check{\rho}(y)\})|$. Then, a matroid $M(\check{\rho})$ is defined by letting S be independent if and only if $\mu_S^\intercal \chi_X \leq \check{\rho}(X)$ for all $X \subseteq \Sigma$. The resulting rank function is,

$$r_{M(\check{\rho})}(S) \triangleq \min_{X \subseteq \Sigma} \check{\rho}(X) + \sum_{y \in \Sigma \setminus X} \mu_S(y),$$

which we note can be evaluated efficiently by submodular minimization algorithms [28, 42, 18, 36] over $X \mapsto \check{\rho}(X) + \sum_{y \in \Sigma \setminus X} \mu_S(y)$. This is a standard construction; see [16, 3] or Ch. 44 of [43] for more discussion.

Algorithm 2 Scaffolding with Weights Computed from Continuous Greedy.

```
1  S ← continuous greedy with rounding on T ↦ Σ_{k=1}^{Δ} f(T(k)) over (E_ρ^∨, M(ρ^∨))
2  forall k ∈ {1, ..., Δ} do
3  |  select any y_1 y_2 ... ∈ S(k)!
4  |  forall i ∈ {1, ..., |S(k)|} do
5  |  |  w^(k)(y_i) ← (f/{y_1, ..., y_{i-1}})(y_i)
6  |  forall z ∉ S(k) do
7  |  |  w^(k)(z) ← 0
8  w̄ ← Δ^{-1} · Σ_{k=1}^{Δ} w^(k)
9  α ← scaffolded greedy on L(M, ⇝) with weights w̄        // Using Algorithm 1
10 return α
```

Now, let $f : 2^{\Sigma} \to \mathbb{R}$ be a monotone submodular function. Our next algorithm is based off of solving a monotone submodular maximization, using a derived objective function $g : 2^{E_\rho^\vee} \to \mathbb{R}$, over the natural matroid using the continuous greedy algorithm [5]. The corresponding solution then defines an additive function underestimating the objective f, which we feed into Algorithm 1 to obtain a word from the MPS. To define g, for $S \subseteq E_\rho^\vee$ let its kth slice $S(k)$ be $S(k) \triangleq \{y \in \Sigma \mid (y,k) \in S\}$. Notice, as Δ is the maximum rank of a principal ideal and $\rho^\vee = r_{\mathcal{M}} \circ \downarrow$, it follows there are at most Δ slices to evaluate for any $S \subseteq E_\rho^\vee$. Then, define $g : 2^{E_\rho^\vee} \to \mathbb{R} : S \mapsto \sum_{k=1}^{\Delta} f(S(k))$. Note g is monotone submodular since it is a sum of monotone submodular functions on disjoint restricted domains.

Algorithm 2 begins by obtaining an approximate solution S to maximizing g over the natural matroid $(E_\rho^\vee, M(\rho^\vee))$ in line 1. Then, in lines 3 through 7 we compute a set of weights for each slice of S. Specifically, an arbitrary permutation $y_1 y_2 \ldots$ is chosen from $S(k)!$ to make $w^{(k)}(y_i) = (f/\{y_1, \ldots, y_{i-1}\})(y_i)$ for all $i \in \{1, \ldots, |S(k)|\}$ and $w^{(k)}(z) = 0$ for $z \notin S(k)$. These weight functions are averaged together in line 8 to obtain the weight function $\overline{w}$. This is, in some sense, an additive estimate of f informed by a solution obtained from a proxy function. We then obtain a word of the MPS in line 9 by running Algorithm 1 with the weights defined by $\overline{w}$, and output that word. The resulting guarantee is that of continuous greedy with pipage rounding [5] degraded by a Δ^2-factor due to the maneuvers involved in translating its solution to a feasible word.

▶ **Theorem 13.** *Let $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ be a matroidal prerequisite system, $f : 2^{\Sigma} \to \mathbb{R}$ a (normalized) monotone submodular function, and* $\mathrm{OPT} = \max_{\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)} f(\widetilde{\alpha})$. *Algorithm 2 outputs $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ with $\mathbb{E}[f(\widetilde{\alpha})] \geq \Delta^{-2}(1 - 1/e - \delta) \cdot \mathrm{OPT}$ in randomized* $\mathsf{poly}(|\Sigma|, 1/\delta)$*-time for $\delta > 0$.*

Proof. Let S be the independent set of $(E_\rho^\vee, M(\rho^\vee))$ approximately maximizing g, as computed in line 1. Then, $\overline{w}^\intercal \mu_S = \Delta^{-1} \cdot \sum_{k=1}^{\Delta} \sum_{y \in \Sigma} \mu_S(y) w^{(k)}(y)$. But, because lines 6-9 make $\sum_{y \in S(k)} w^{(k)}(y) = f(S(k))$, we find,

$$\frac{1}{\Delta} \cdot \sum_{k=1}^{\Delta} \sum_{y \in \Sigma} \mu_S(y) w^{(k)}(y) \geq \frac{1}{\Delta} \cdot \sum_{k=1}^{\Delta} \sum_{y \in S(k)} w^{(k)}(y) \geq \frac{1}{\Delta} \cdot \sum_{k=1}^{\Delta} f(S(k)) = \frac{g(S)}{\Delta},$$

hence $\overline{w}^\intercal \mu_S \geq \Delta^{-1} \cdot g(S)$.

Now, let $v^\star$ be the solution computed by Edmonds' algorithm over the base polytope of $\rho^\vee$ when given weights $\overline{w}$, and examine the final word α computed in line 9. Because no coordinate in $v^\star$ can exceed value Δ, $\sum_{y \in \widetilde{\alpha}} \overline{w}(y) \geq \Delta^{-1} \overline{w}^\intercal v^\star$ follows by the law of

Algorithm 3 Marginal Adjusted Greedy for Additive Maximization.

1 $\alpha \leftarrow \epsilon$
2 **while** $|\alpha| < r_{\mathcal{M}}(\Sigma)$ **do**
3 $\quad \ell \leftarrow |\alpha|$
4 $\quad x_1 \dots x_\ell \leftarrow \alpha$
5 $\quad K_0 \leftarrow \varnothing$
6 $\quad$ **for** $i \in \{1, \dots, \ell\}$ **do** $K_i \leftarrow \mathrm{cl}(\{x_1, \dots, x_i\})$
7 $\quad$ **for** $i \in \{0, \dots, \ell\}$ **do**
8 $\quad\quad$ **if** $i < \ell$ **then** $\Psi_i \leftarrow \{y \in \Sigma \mid K_i \subsetneq \mathrm{cl}(K_i + y) \subsetneq K_{i+1}\}$
9 $\quad\quad$ **else** $\Psi_i \leftarrow \{y \in \Sigma \mid K_i \subsetneq \mathrm{cl}(K_i + y) \subseteq \Sigma\}$
10 $\quad (y^\star, i^\star) \leftarrow \underset{y \in \Psi_i : 0 \le i \le \ell}{\arg\max} \frac{w(y)}{(\overset{\vee}{\rho}/K_i)(y)}$
11 $\quad \alpha \leftarrow x_1 \cdots x_{i^\star}\, y^\star\, x_{i^\star+1} \cdots x_\ell$
12 **return** α

diminishing returns. But, the natural matroid construction shows μ_S is in the base polytope, and so by optimality of $v^\star$ we see $\overline{w}^\intercal \mu_S \le \overline{w}^\intercal v^\star$. So, combining this with the insight of the last paragraph $\overline{w}^\intercal v^\star/\Delta \ge \overline{w}^\intercal \mu_S/\Delta \ge g(S)/\Delta^2$. Then, noting that $\sum_{y \in X} w^{(k)}(y) \le f(X)$ for all $k \in \{1, \dots, \Delta\}$ and $X \subseteq \Sigma$ by the law of diminishing returns, combining the last equation with $\sum_{y \in \widetilde{\alpha}} \overline{w}(y) \ge \Delta^{-1} \overline{w}^\intercal v^\star$ shows $f(\widetilde{\alpha}) \ge \sum_{y \in \widetilde{\alpha}} \overline{w}(y) \ge \Delta^{-2} g(S)$. To finish things up, let α_{OPT} be an optimal basic word, and $T_{\alpha_{\mathrm{OPT}}} = \{(y, 1) \mid y \in \widetilde{\alpha}_{\mathrm{OPT}}\}$. See that $g(T_{\alpha_{\mathrm{OPT}}}) = f(\widetilde{\alpha}_{\mathrm{OPT}}) + \sum_{k=2}^{\Delta} f(\varnothing) = f(\widetilde{\alpha}_{\mathrm{OPT}}) = \mathrm{OPT}$. So, combining this with the guarantees of [5] and $f(\widetilde{\alpha}) \ge \Delta^{-2} g(S)$ gives $\mathbb{E}[f(\widetilde{\alpha})] \ge \Delta^{-2} \mathbb{E}[g(S)] \ge \Delta^{-2}(1 - 1/e - \delta) \cdot \mathrm{OPT}$, which is the desired result. ◀

6 Marginal Adjusted Greedy Algorithms

The last algorithms were based upon "scaffolding" the solution with good letters from a relaxed solution, and using the fill-in procedure from Lemma 11 to project this solution to a characteristic vector corresponding to a basic word. However, the fill-in procedure was such that the letters were chosen only for feasibility, not with any heuristic guiding their value for a solution. In the next algorithms, we instead select letters using a greedy heuristic driven by the greatest representation. These algorithms will lead to approximation guarantees varying in the largest matroid connectivity $\lambda_{\max}$.

6.1 Additive Maximization

Algorithm 3 works by iteratively selecting letters to insert into a working solution (initially the empty word) in a way that both achieves good value and converges to a feasible word. In each round, the method first computes in line 6 the closed set K_i given by each prefix $x_1 \dots x_i$ of the current value of α, which we assume to be of length ℓ. We also make $K_0 = \varnothing$. Then, in line 8, for each $i \in \{0, 1, \dots, \ell\}$ we compute the set Ψ_i consisting of letters which strictly increase the closed set spanned by K_i to a closed set strictly less than K_{i+1}. The set Ψ_ℓ poses a boundary condition which is handled in line 9. We save such letters in the set Ψ_i because they can be inserted into α between x_i and x_{i+1} (or appended onto the beginning or end of α when i equals 0 or ℓ, respectively) while maintaining the invariant that the process will converge to a feasible word by way of Lemma 11. In particular, one can

view α as consisting of decisions we have already committed to (along with their relative ordering). Under this perspective, we use the closure structure and construction of the sets $\{\Psi_i\}_i$ to identify new decisions we can feasibly make while still preserving the structure of the decisions we have already committed to. This is shown in a technical way in Eq. 11 in the proof of Theorem 14. Finally, observe each letter is a member of at most one Ψ_i, because $y \in \Psi_i$ implies $y \in K_{i+1} \setminus K_i$ and the sequence $\{K_i\}_i$ forms a filtration.

Keeping this in mind, in line 10 we compute a marginal-adjusted weight for every letter $y \in \bigcup_{i=0}^{\ell} \Psi_i$. Specifically, when $y \in \Psi_i$, we divide $w(y)$ by the marginal return $(\check{\rho}/K_i)(y)$. This is because $(\check{\rho}/K_i)(y)$ is exactly the difference in grade between K_i and $\mathrm{cl}(K_i + y)$, and so a large marginal return corresponds to a large jump in the closed set structure. If we commit to selecting y, then we commit to selecting a sequence of letters making a saturated chain between K_i and $\mathrm{cl}(K_i + y)$. This is, the larger the size of this jump, the more we constrain the set of letters we can feasibly select in future rounds to insert into the solution. So, we divide the weight by $(\check{\rho}/K_i)(y)$ to penalize letters which induce large jumps in the closure structure. Then, in line 10, we select the letter $y^\star$ with the largest marginal adjusted weight, and save the index $i^\star$ with $y^\star \in \Psi_{i^\star}$. We then insert $y^\star$ into α between the letters $x_{i^\star}$ and $x_{i^\star+1}$ (or place onto the beginning or end of α when i equals 0 or ℓ respectively), and repeat until $|\alpha|$ equals the length of a basic word in $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$.

For the analysis, let α_{OPT} be an optimal word. We need to "charge" each letter in the optimal word to a decision made by the algorithm. So, first let $y_t^\star$ be the letter selected in round t and $K_t^\star$ be $K_{i^\star}$ from that round. Also, let $R = \check{\rho}(\Sigma)$. By optimism, for all letters $z \in \Sigma$, there exists $t \in \{1, \ldots, R\}$ such that $z \in \mathrm{cl}(K_t^\star + y_t^\star) \setminus K_t^\star$. Then, let L_t be the letters from $\widetilde{\alpha}_{\mathrm{OPT}}$ for which t is the last round this occurs. Formally, this is given by the recursion over $t \in \{1, \ldots, R\}$,

$$L_t \triangleq \left(\mathrm{cl}(K_t^\star + y_t^\star) \setminus K_t^\star\right) \setminus \left(\bigcup_{t'=t+1}^{R} L_t\right), \quad \text{with } L_R = \mathrm{cl}(K_R^\star + y_R^\star) \setminus K_R^\star. \tag{8}$$

And, by our prior observation we see $(L_t)_t$ is a partition of $\widetilde{\alpha}_{\mathrm{OPT}}$. This sets up the sum,

$$\sum_{y \in \widetilde{\alpha}_{\mathrm{OPT}}} w(y) = \sum_{t=1}^{R} \left(\sum_{y \in L_t} w(y)\right). \tag{9}$$

But, letting $\delta_t = (\check{\rho}/K_t^\star)(y_t^\star)$, the choice rule makes $w(y) \le \delta_t^{-1} w(y_t^\star)\delta_t \cdot \left(\check{\rho}/K_{i^\star}\right)(y)$ for all $y \in L_t$, while $L_t \subseteq \mathrm{cl}(K_t^\star + y_t^\star) \setminus K_t^\star$ makes $\left(\check{\rho}/K_t^\star\right)(y) \le \left(\check{\rho}/K_t^\star\right)(y_t^\star) = \delta_t$ for all $y \in L_t$. Therefore,

$$\sum_{t=1}^{R} \left(\sum_{y \in L_t} w(y)\right) \le \sum_{t=1}^{R} \left(|L_t| \cdot \frac{w(y_t^\star)}{\delta_t} \cdot \left(\check{\rho}/K_{i^\star}\right)(y)\right) \le \sum_{t=1}^{R} |L_t| \cdot w(y_t^\star). \tag{10}$$

So, noting that $\sum_{t=1}^{R} w(y_t^\star)$ is the value of the solution returned by Algorithm 3, we need only obtain a uniform bound on $|L_t|$ for an approximation guarantee. This is our approach. Recall from Section 1 that $\lambda_{\max}$ is the maximum value of the matroid connectivity function of the matroid defining the MPS (see Eq. 2). A careful analysis will show $|L_t| \le 1 + \lambda_{\max}$.

▶ **Theorem 14.** *Let $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ be a matroidal prerequisite system, $w : \Sigma \to \mathbb{R}_+$ a nonnegative weight function, and* $\mathrm{OPT} = \max_{\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)} \sum_{y \in \widetilde{\alpha}} w(y)$. *Algorithm 3 outputs $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ with $\sum_{y \in \widetilde{\alpha}} w(y) \ge (1 + \lambda_{\max})^{-1} \cdot \mathrm{OPT}$.*

Proof. Firstly, see that α is feasible because lines 8 and 9 in Algorithm 3 ensure the following invariant: the evaluations of cl on each prefix of the working solution forms a strictly increasing chain in $(\kappa(\Lambda)/\!\sim)$. Since $\sigma^{\vee}_{\rho} = \mathrm{cl}$, it can be seen then that the working solution satisfies the premise of Lemma 11 at all times. Hence, once it achieves the length of a basic word, it must be feasible.

Now, before continuing, observe the letters of the solution α are not arranged in the order in which they were inserted. In light of this, let $\pi : \{1, \ldots, R\} \to \{1, \ldots, R\}$ map the index corresponding to the position of a letter in α to the round index it was added to the solution. Then, we may write $\alpha = y^{\star}_{\pi(1)} \ldots y^{\star}_{\pi(R)}$ while still identifying the round a letter was added. Furthermore, also define $X_i \triangleq \left\{y^{\star}_{\pi(1)}, \ldots, y^{\star}_{\pi(i)}\right\}$ as the first i letters in the algorithm's solution α. Take note that as $y^{\star}_{\pi(1)}, \ldots, y^{\star}_{\pi(0)} = \epsilon$ by convention, it follows that $X_0 = \varnothing$. Moreover, the insertion scheme of Algorithm 3 maintains,

$$\mathrm{cl}(X_i) = \mathrm{cl}\left(K^{\star}_{\pi(i)} + y^{\star}_{\pi(i)}\right), \quad \forall i \in \{1, \ldots, R\}. \tag{11}$$

To see why, fix $i \in \{1, \ldots, R\}$ and select an index j minimizing $\pi(j)$ among $j < i$ making $\pi(j) > \pi(i)$; this is, $y^{\star}_{\pi(j)}$ is the first letter preceding $y^{\star}_{\pi(i)}$ in position in α which was selected for insertion in a later round than $y^{\star}_{\pi(i)}$ was selected. Note that if no such j exists, then Eq. 11 holds as $K^{\star}_{\pi(i)} = \mathrm{cl}(X_{i-1})$ would have to be true. Then, $y^{\star}_{\pi(j)}$ belongs to a set in the sequence $(\Psi_{i'})_{i'}$ computed in line 8, and because $y^{\star}_{\pi(j)}$ precedes $y^{\star}_{\pi(i)}$ in position in α and minimizes $\pi(j)$ amongst such letters, this set from $(\Psi_{i'})_{i'}$ must be a subset of,

$$\mathrm{cl}\left(\left\{y^{\star}_{\pi(k)} \mid \pi(k) \leq \pi(i) \text{ and } k \leq i\right\}\right) = \mathrm{cl}\left(K^{\star}_{\pi(i)} + y^{\star}_{\pi(i)}\right). \tag{12}$$

So, $y_{\pi(j)} \in \mathrm{cl}\left(K^{\star}_{\pi(i)} + y^{\star}_{\pi(i)}\right)$. Now, to induct, select an arbitrary index j' with $j' < i$ and $\pi(j') > \pi(j)$. This is, $y^{\star}_{\pi(j')}$ is any letter preceding $y^{\star}_{\pi(i)}$ in position in α which was selected for insertion in a later round than $y^{\star}_{\pi(i)}$ and $y^{\star}_{\pi(j)}$. By inductive hypothesis,

$$k < i \text{ and } \pi(i) < \pi(k) < \pi(j') \implies y^{\star}_{\pi(k)} \in \mathrm{cl}\left(K^{\star}_{\pi(i)} + y^{\star}_{\pi(i)}\right).$$

Combining this with Eq. 12 shows $\mathrm{cl}\left(\left\{y^{\star}_{\pi(k)} \mid \pi(k) < \pi(j') \text{ and } k \leq i\right\}\right) = \mathrm{cl}\left(K^{\star}_{\pi(i)} + y^{\star}_{\pi(i)}\right)$. The left hand side is the closure of the support of the prefix of the working solution ending at $y^{\star}_{\pi(i)}$ in round $\pi(j')$. Then, because $j' < i$, i.e. $y^{\star}_{\pi(j')}$ precedes $y^{\star}_{\pi(i)}$ in position in α, line 8 implies $y^{\star}_{\pi(j')} \in \mathrm{cl}\left(K^{\star}_{\pi(i)} + y^{\star}_{\pi(i)}\right)$ as well. So, by the principle of induction we see $X_{i-1} \subseteq \mathrm{cl}\left(K^{\star}_{\pi(i)} + y^{\star}_{\pi(i)}\right)$, which by $X_{i-1} + y^{\star}_{\pi(i)} = X_i$ and idempotency of closure operators implies $\mathrm{cl}(X_i) \subseteq \mathrm{cl}\left(K^{\star}_{\pi(i)} + y^{\star}_{\pi(i)}\right)$. Of course, the reverse inclusion follows from,

$$\begin{aligned}
\mathrm{cl}\left(K^{\star}_{\pi(i)} + y^{\star}_{\pi(i)}\right) &= \mathrm{cl}\left(\mathrm{cl}\left(\left\{y^{\star}_{\pi(k)} \mid k < i \text{ and } \pi(k) < \pi(i)\right\}\right) + y^{\star}_{\pi(i)}\right), \\
&= \mathrm{cl}\left(\left\{y^{\star}_{\pi(k)} \mid k < i \text{ and } \pi(k) < \pi(i)\right\} + y^{\star}_{\pi(i)}\right),
\end{aligned}$$

which is a subset of $\mathrm{cl}(X_i)$ by the monotonicity of closure operators. And so, Eq. 11 follows.

With that done, the approximation guarantee will follow by proving the following,

$$z \in L_{\pi(i)} \implies z \in \mathrm{cl}\left(X_i\right) \setminus \mathrm{cl}\left(X_{i-1}\right). \tag{13}$$

To see why, observe $L_{\pi(i)} \subseteq \mathrm{cl}\left(K^{\star}_{\pi(i)} + y^{\star}_{\pi(i)}\right) \setminus K^{\star}_{\pi(i)}$. So, by way of contradiction, suppose Eq. 13 does not hold. There exists $z \in L_{\pi(i)}$ with $z \notin \mathrm{cl}\left(X_i\right) \setminus \mathrm{cl}\left(X_{i-1}\right)$. As

$z \in \operatorname{cl}\left(K^\star_{\pi(i)} + y^\star_{\pi(i)}\right) \setminus K^\star_{\pi(i)}$ and $\operatorname{cl}\left(K^\star_{\pi(i)} + y^\star_{\pi(i)}\right) \subseteq \operatorname{cl}(X_i)$, since $K^\star_{\pi(i)} + y^\star_{\pi(i)}$ is the union of the cl-closure of a subset of X_{i-1} unioned with the ith letter in α, for Eq. 13 to not hold it must be that $z \in \operatorname{cl}(X_{i-1})$. This shows,

$$\left(\exists z \in L_{\pi(i)}\right)\ z \in \operatorname{cl}\left(X_{i-1}\right) \setminus K^\star_{\pi(i)}. \tag{14}$$

However, $\operatorname{cl}(X_{i-1}) = \operatorname{cl}\left(K^\star_{\pi(i-1)} + y^\star_{\pi(i-1)}\right)$ by Eq. 11, so by $z \in L_{\pi(i)}$ it must be that $\pi(i-1) < \pi(i)$. This means that $y^\star_{\pi(i-1)}$ was in the support of the working solution when $y^\star_\pi(i)$ was inserted. Since $y^\star_{\pi(i-1)}$ directly precedes $y^\star_\pi(i)$ in position in α, this implies $K^\star_{\pi(i)} = \operatorname{cl}\left(K^\star_{\pi(i-1)} + y^\star_{\pi(i-1)}\right)$. But $\operatorname{cl}(X_{i-1}) = \operatorname{cl}\left(K^\star_{\pi(i-1)} + y^\star_{\pi(i-1)}\right)$ implies $\operatorname{cl}\left(X_{i-1}\right) = K^\star_{\pi(i)}$, which contradicts Eq. 14. So, we find Eq. 13 holds true.

As a result, we observe $L_t \subseteq \widetilde{\alpha}_{\text{OPT}} \cap \left(\operatorname{cl}\left(X_i\right) \setminus \operatorname{cl}\left(X_{i-1}\right)\right)$. Now, we make use of the matroid structure to introduce $\lambda_{\max}$. Because $\widetilde{\alpha}_{\text{OPT}}$ is a basic set in $\mathcal{M}$ (see Section 4),

$$\left|\widetilde{\alpha}_{\text{OPT}} \cap \operatorname{cl}\left(X_{i-1}\right)\right| = r_{\mathcal{M}}(\Sigma) - r_{\mathcal{M}}(\widetilde{\alpha}_{\text{OPT}} \setminus \operatorname{cl}\left(X_{i-1}\right)). \tag{15}$$

Continuing on, note that because $\operatorname{cl}(X_i)$ is the smallest flat support containing $y^\star_1 \dots y^\star_i$, it follows that $\operatorname{cl}(X_i) = \kappa[y^\star_1 \dots y^\star_i]$. This logic also shows $\operatorname{cl}(X_{i-1}) = \kappa[y^\star_1 \dots y^\star_{i-1}]$. Then, $\operatorname{cl}(X_i) = \sigma_{\mathcal{M}}(\{y^\star_1, \dots, y^\star_i\})$ and $\operatorname{cl}(X_{i-1}) = \sigma_{\mathcal{M}}(\{y^\star_1, \dots, y^\star_{i-1}\})$ by Lemma 7.1. Because feasible words in the MPS are independent in $\mathcal{M}$, we have $r_{\mathcal{M}}(\operatorname{cl}(X_i)) = r_{\mathcal{M}}(\operatorname{cl}(X_{i-1})) + 1$. Since independence implies $|\widetilde{\alpha}_{\text{OPT}} \cap \operatorname{cl}(X_i)| \leq r_{\mathcal{M}}(\operatorname{cl}(X_i))$, we see $|\widetilde{\alpha}_{\text{OPT}} \cap \operatorname{cl}(X_i)| \leq r_{\mathcal{M}}(\operatorname{cl}(X_{i-1})) + 1$. This observation combines with Eq. 15 for,

$$\begin{aligned}
|L_t| &\leq |\widetilde{\alpha}_{\text{OPT}} \cap (\operatorname{cl}(X_i) \setminus \operatorname{cl}(X_{i-1}))\,|,\\
&= |\widetilde{\alpha}_{\text{OPT}} \cap \operatorname{cl}(X_i)| - |\widetilde{\alpha}_{\text{OPT}} \cap \operatorname{cl}(X_{i-1})|,\\
&\leq 1 + r_{\mathcal{M}}(\operatorname{cl}(X_{i-1})) + r_{\mathcal{M}}(\widetilde{\alpha}_{\text{OPT}} \setminus \operatorname{cl}(X_{i-1})) - r_{\mathcal{M}}(\Sigma),\\
&\leq 1 + \left(r_{\mathcal{M}}(\operatorname{cl}(X_{i-1})) + r_{\mathcal{M}}(\Sigma \setminus \operatorname{cl}(X_{i-1})) - r_{\mathcal{M}}(\Sigma)\right),
\end{aligned}$$

which is less than or equal to $1 + \lambda_{\max}$ by definition (see Eq. 2). And so, by Eq.s 9 and 10 it is true that $\text{OPT} \leq \sum_{t=1}^{R} |L_t| \cdot w(y^\star_t) \leq (1 + \lambda_{\max}) \cdot \sum_{t=1}^{R} w(y^\star_t)$, which shows α is a $(1 + \lambda_{\max})$-approximation of the optimal solution. ◀

6.2 Monotone Submodular Maximization

We can extend this method to submodular maximization by modifying the selection heuristic: the only change from Algorithm 3 is that we compute the adjusted weight of a letter y in line 10 using the marginal return $(f/\widetilde{\alpha})(y)$ under f against the current working solution instead of the weight $w(y)$. We reuse the previous analysis to obtain the following.

▶ **Theorem 15.** *Let $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ be a matroidal prerequisite system, $f : 2^\Sigma \to \mathbb{R}$ a (normalized) monotone submodular function, and $\text{OPT} = \max_{\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)} f(\widetilde{\alpha})$. After modifying line 10 of Algorithm 3 to compute the marginal adjusted weight $(f/\widetilde{\alpha})(y)/(\overset{\vee}{\rho}/K_i)(y)$, the algorithm outputs α with $f(\widetilde{\alpha}) \geq (2 + \lambda_{\max})^{-1} \cdot \text{OPT}$.*

Proof. Our analysis reuses parts of that presented for Algorithm 3. Notably, α is feasible by way of Lemma 11. Continuing on, define L_t in the same way as Eq. 8. Notice that the argument bounding $|L_t|$ presented in the proof of Theorem 14 only uses properties of the MPS and not the additive objective function, so we have $|L_t| \leq 1 + \lambda_{\max}$. Also let $y^\star_t$ be the

letter selected in round t, α_{OPT} and $K_t^\star$ defined as before, as well as $\alpha^{(t)}$ be the value of the working solution α at the end of round t (with the convention $\alpha^{(0)} = \epsilon$). Now, our choice rule makes,

$$\left(f/\widetilde{\alpha}^{(t-1)}\right)(z) \leq \frac{\left(f/\widetilde{\alpha}^{(t-1)}\right)(y_t^\star)}{(\overset{\vee}{\rho}/K_t^\star)(z)}(\overset{\vee}{\rho}/K_t^\star)(y_t^\star) \leq \left(f/\widetilde{\alpha}^{(t-1)}\right)(y_t^\star), \quad \forall z \in L_t,$$

and so we find that for all $t \in \{1, \ldots, R\}$,

$$\sum_{z \in L_t} \left(f/\widetilde{\alpha}^{(t-1)}\right)(z) \leq |L_t| \cdot \left(f/\widetilde{\alpha}^{(t-1)}\right)(y_t^\star) \leq (1 + \lambda_{\max}) \cdot \left(f/\widetilde{\alpha}^{(t-1)}\right)(y_t^\star).$$

So, because $f(\widetilde{\alpha}) = \sum_{t=1}^{R} \left(f/\widetilde{\alpha}^{(t-1)}\right)(y_t^\star)$, summing over all rounds shows,

$$\sum_{t=1}^{R} \sum_{z \in L_t} \left(f/\widetilde{\alpha}^{(t-1)}\right)(z) \leq (1 + \lambda_{\max}) \cdot f(\widetilde{\alpha}). \tag{16}$$

We relate the left-hand side to OPT via $\mathrm{OPT} \leq f(\widetilde{\alpha} \cup \widetilde{\alpha}_{\mathrm{OPT}}) \leq f(\widetilde{\alpha}) + \sum_{z \in \widetilde{\alpha}_{\mathrm{OPT}} \setminus \widetilde{\alpha}} (f/\widetilde{\alpha})(z)$, as the first inequality follows by monotonicity, while the second follows by the law of diminishing returns. Well, by the law of diminishing returns we also have $z \in L_t$ implies $\left(f/\widetilde{\alpha}^{(t-1)}\right)(z) \geq (f/\widetilde{\alpha})(z)$ since $\widetilde{\alpha}^{(t-1)} \subseteq \widetilde{\alpha}$ always holds by our construction. This and Eq. 16 implies $\mathrm{OPT} \leq f(\widetilde{\alpha}) + \sum_{t=1}^{R} \sum_{z \in L_t} \left(f/\widetilde{\alpha}^{(t-1)}\right)(z) \leq (2 + \lambda_{\max}) \cdot f(\widetilde{\alpha})$. Rearranging terms gives the desired result. ◀

7 Hardness of Approximation of Additive Maximization

Finally, we show that a $\min\{\Delta, \lambda_{\max}\}^{o(1)}$-approximation for nonnegative additive maximization over an MPS is intractable. We reduce from *Densest k-Subgraph (Dk-S)*: Given an undirected graph (V, E) and integer k, select a vertex subset $S \subseteq V$ of size $|S| = k$ maximizing the induced edges. The Gap Exponential Time Hypothesis (Gap-ETH) conjectures the existence of a small $\delta > 0$ for which no subexponential time algorithm can distinguish between a satisfiable 3-SAT formula and one where at most a $(1 - \delta)$-fraction of clauses can be satisfied [30, 9, 6]. Under the Gap-ETH, there is no $|V|^{o(1)}$-approximation to Dk-S [29].

For a coming arithmetic exercise, define $B = |E| + 1$. We construct an MPS whose nonnegative additive maximization solutions encode Dk-S solutions in a way preserving their approximation quality. For each $u \in V$ make B "dummy" letters $d_{u,1}, \ldots, d_{u,B}$, include each vertex u, and include each edge $e = \{u, v\} \in E$. For the weights set,

$$w(y) = \begin{cases} 1, & \text{if } y \text{ is an edge letter,} \\ 0, & \text{otherwise,} \end{cases} \tag{17}$$

and let the matroid $\mathcal{M}$ be the uniform matroid of rank $R \triangleq k(B+1) + |E| + 1$. Finally, for the prerequisite order, each vertex u is greater than all its dummy letters $d_{u,1}, \ldots, d_{u,B}$ and each edge $e = \{u, v\} \in E$ is greater than u and v. Recall, we showed in Section 4 that uniform matroids always satisfy compatibility with $(\Sigma, \rightsquigarrow)$. So, $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$ is a MPS.

We describe how the solutions values of the MPS and Dk-S instances are related in the following two claims. This is then used to prove Lemma 18, which gives an approximation-preserving reduction from Dk-S to nonnegative additive maximization over an MPS.

▷ Claim 16. There is an efficient procedure to extract a k-subgraph with at least $\mathrm{VAL} - 1$ edges from every basic word of cumulative weight VAL.

Proof. Let $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$ be a basic word of value VAL, and let α' be the prefix excluding the last letter. We exclude the last letter because it can correspond to an edge which was added without both its incident vertices being present in the word. However, because the matroid is uniform, it is not possible for the prerequisites of any edge in $\widetilde{\alpha}'$ to be satisfied "virtually" without their membership in the support. Therefore, because the edges correspond to the only letters with nonzero weight, if we let the cumulative weight of the support of α' be VAL$'$ then the vertices in $\widetilde{\alpha}'$ must induce a subgraph with at least VAL$'$ edges.

Then, examine the set $X = \{v \in V \mid (\exists e \in \widetilde{\alpha}' \cap E)\ v \in e\}$. We claim $|X| \leq k$. By way of contradiction, suppose this is not true. Our prior observations make $X \subseteq \widetilde{\alpha}'$. Note then that for every letter in X, the B dummy prerequisite letters must also be present. So, if $|X| > k+1$ it follows that there are at least $k+1$ vertex letters and $B(k+1)$ dummy letters in $\widetilde{\alpha}'$. By recalling $B = |E|+1$, one observes $|\widetilde{\alpha}'| \geq (k+1)(B+1) = k(B+1) + |E| + 2 > R$. This means $\widetilde{\alpha}'$ is not independent, which contradicts the fact that it is the prefix of a basic word of $\mathcal{L}(\mathcal{M}, \rightsquigarrow)$. And so, because $|X| \leq k$, one can output a feasible Dk-S solution of value at least VAL$'$ by listing the vertices in X along with a choice of $k - |X|$ additional vertices. Since VAL$' \geq$ VAL $- 1$, this shows the claim. ◁

▷ Claim 17. For any Dk-S solution inducing VAL edges, there exists a basic word of cumulative weight at least VAL.

Proof. Let X be a Dk-S solution of value VAL. Take $S = \{(u,v) \in E \mid \{u,v\} \subseteq X\}$ as the edges in the induced subgraph; clearly $|S| =$ VAL. Now, examine a word α formed by taking any linear extension of the ideal generated by S in $(\Sigma, \rightsquigarrow)$. Observe, $|X| = k$ implies there can be at most k vertex letters in this ideal, and each vertex letter is above exactly B letters associated with its dummies. Thus $|\alpha| \leq k(B+1) + (\text{VAL}) \leq k(B+1) + |E|$, and so $\widetilde{\alpha}$ is independent. Furthermore, the prerequisite conditions are trivially satisfied by α since it is a linear extension of an ideal (i.e. for any $y \in \widetilde{\alpha}$, *every* $x \rightsquigarrow y$ is such that $x \in \widetilde{\alpha}$ in a position occurring before the appearance of y), so $\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)$. Since the support of α contains VAL edges, its cumulative weight is at least VAL. Therefore, any basic word formed by concatenating α with a suffix from $\mathcal{L}(\mathcal{M}, \rightsquigarrow)/[\alpha]$ gives the result. ◁

▶ **Lemma 18.** *Fix a Dk-S instance. Let Σ, $\mathcal{M}$, and $(\Sigma, \rightsquigarrow)$ as constructed, and $w : \Sigma \to \mathbb{R}$ be as defined in Eq. 17. Let $\text{OPT}_{Dk\text{-}S}$ be the number of edges in an optimal Dk-S solution, and $\text{OPT}_{MPS} = \max_{\alpha \in \mathcal{L}(\mathcal{M}, \rightsquigarrow)} \sum_{y \in \widetilde{\alpha}} w(y)$. If there exists an efficient δ^{-1}-approximation algorithm to nonnegative additive maximization over the MPS, then there exists an efficient algorithm giving a Dk-S solution inducing $\delta \cdot (\text{OPT}_{Dk\text{-}S} + O(1)) - 1$ edges.*

Proof. Let OPT_{MPS} be the weight of an optimal basic word. The first claim we proved shows $\text{OPT}_{\text{MPS}} \leq \text{OPT}_{\text{D}k\text{-S}} + 1$ while the second shows $\text{OPT}_{\text{D}k\text{-S}} \leq \text{OPT}_{\text{MPS}}$. So, $\text{OPT}_{\text{MPS}} = \text{OPT}_{\text{D}k\text{-S}} + O(1)$. Moreover, the first claim also shows an efficient procedure for extracting a Dk-S solution of value $\delta \cdot \text{OPT}_{\text{MPS}} - 1$ from any δ-approximation to the nonnegative additive maximization over the MPS. Altogether, $\delta \cdot \text{OPT}_{\text{MPS}} - 1 = \delta \cdot (\text{OPT}_{\text{D}k\text{-S}} + O(1)) - 1$. ◀

Proof of Main Result 2. Now, observe the principal ideal of any edge letter is a maximum sized such ideal with cardinality exactly $2 \cdot (B+1) + 1$. Since $R \geq 2 \cdot (B+1)$ for $k \geq 1$, this means for any choice of edge e we have $\Delta = r_{\mathcal{M}}(\downarrow e) = \Theta(|E|) = O(|V|^2)$. Therefore, by Lemma 18, any $\Delta^{o(1)}$-approximation algorithm for an MPS computes a $|V|^{o(1)}$-approximation to Dk-S. Moreover, one can observe that $\lambda_{\max} \leq r_{\mathcal{M}}(\Sigma)$ is always true. And so, $\lambda_{\max} \leq R = O(|E|) = O(|V|^2)$. Thus, Lemma 18 also implies any $\lambda_{\max}^{o(1)}$-approximation algorithm for an MPS is a $|V|^{o(1)}$-approximation to Dk-S. ◀

References

1 Caterina Bigoni, Zhenying Zhang, and Jan S Hesthaven. Systematic sensor placement for structural anomaly detection in the absence of damaged states. *Computer Methods in Applied Mechanics and Engineering*, 371:113315, 2020.

2 Anders Björner and Günter M Ziegler. Introduction to greedoids. *Matroid applications*, 40:284–357, 1992.

3 Joseph E Bonin, Carolyn Chun, and Tara Fife. The natural matroid of an integer polymatroid. *SIAM Journal on Discrete Mathematics*, 37(3):1751–1770, 2023. doi:10.1137/22M1521122.

4 Niv Buchbinder and Moran Feldman. Deterministic algorithm and faster algorithm for submodular maximization subject to a matroid constraint. *SIAM Journal on Computing*, pages FOCS24–85, 2025.

5 Gruia Calinescu, Chandra Chekuri, Martin Pal, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011. doi:10.1137/080733991.

6 Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From gap-eth to fpt-inapproximability: Clique, dominating set, and more. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 743–754. IEEE, 2017. doi:10.1109/FOCS.2017.74.

7 Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 575–584. IEEE, 2010. doi:10.1109/FOCS.2010.60.

8 Asaf Cohen, Shlomi Dolev, and Guy Leshem. Sensor networks: From dependence analysis via matroid bases to online synthesis. *Theoretical Computer Science*, 553:2–17, 2014. doi:10.1016/J.TCS.2014.07.014.

9 Irit Dinur. Mildly exponential reduction from gap-3sat to polynomial-gap label-cover. In *Electronic colloquium on computational complexity ECCC; research reports, surveys and books in computational complexity*, page 128, 2016.

10 Jack Edmonds. Matroids and the greedy algorithm. *Mathematical Programming*, 1:127–136, 1971. doi:10.1007/BF01584082.

11 Jack Edmonds. Submodular functions, matroids, and certain polyhedra. In *Combinatorial Optimization—Eureka, You Shrink! Papers Dedicated to Jack Edmonds 5th International Workshop Aussois, France, March 5–9, 2001 Revised Papers*, pages 11–26. Springer, 2003.

12 Ulrich Faigle. On ordered languages and the optimization of linear functions by greedy algorithms. *Journal of the ACM (JACM)*, 32(4):861–870, 1985. doi:10.1145/4221.4998.

13 Moran Feldman, Joseph Naor, Roy Schwartz, and Justin Ward. Improved approximations for k-exchange systems. In *European Symposium on Algorithms*, pages 784–798. Springer, 2011.

14 Yuval Filmus and Justin Ward. Monotone submodular maximization over a matroid via non-oblivious local search. *SIAM Journal on Computing*, 43(2):514–542, 2014. doi:10.1137/130920277.

15 Roy H Goetschel Jr. Linear objective functions on certain classes of greedoids. *Discrete applied mathematics*, 14(1):11–16, 1986. doi:10.1016/0166-218X(86)90003-X.

16 Thorkell Helgason. Aspects of the theory of hypermatroids. In *Hypergraph Seminar*, pages 191–213. Springer, 1974.

17 Paul Helman, Bernard ME Moret, and Henry D Shapiro. An exact characterization of greedy structures. *SIAM Journal on Discrete Mathematics*, 6(2):274–283, 1993. doi:10.1137/0406021.

18 Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *Journal of the ACM (JACM)*, 48(4):761–777, 2001. doi:10.1145/502090.502096.

19 David S Johnson and KA Niemi. On knapsacks, partitions, and a new dynamic programming technique for trees. *Mathematics of Operations Research*, 8(1):1–14, 1983. doi:10.1287/MOOR.8.1.1.

20 Stavros G Kolliopoulos and George Steiner. Partially ordered knapsack and applications to scheduling. *Discrete Applied Mathematics*, 155(8):889–897, 2007. doi:10.1016/J.DAM.2006.08.006.

21 Bernhard Korte and Dirk Hausmann. An analysis of the greedy heuristic for independence systems. In *Annals of Discrete Mathematics*, volume 2, pages 65–74. Elsevier, 1978.

22 Bernhard Korte and László Lovász. Greedoids-a structural framework for the greedy algorithm. In *Progress in combinatorial optimization*, pages 221–243. Elsevier, 1984.

23 Bernhard Korte and László Lovász. Greedoids and linear objective functions. *SIAM Journal on Algebraic Discrete Methods*, 5(2):229–238, 1984.

24 Bernhard Korte and László Lovász. Polymatroid greedoids. *Journal of Combinatorial Theory, Series B*, 38(1):41–72, 1985. doi:10.1016/0095-8956(85)90091-7.

25 Bernhard Korte, László Lovász, and Rainer Schrader. *Greedoids*, volume 4. Springer Science & Business Media, 2012.

26 Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(2), 2008.

27 Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Mathematics of Operations Research*, 35(4):795–806, 2010. doi:10.1287/MOOR.1100.0463.

28 László Lovász. Submodular functions and convexity. In *Mathematical Programming The State of the Art: Bonn 1982*, pages 235–257. Springer, 1983. doi:10.1007/978-3-642-68874-4_10.

29 Pasin Manurangsi. Almost-polynomial ratio eth-hardness of approximating densest k-subgraph. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 954–961, 2017. doi:10.1145/3055399.3055412.

30 Pasin Manurangsi and Prasad Raghavendra. A Birthday Repetition Theorem and Complexity of Approximating Dense CSPs. In *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*, volume 80 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 78:1–78:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.78.

31 S Thomas McCormick, Britta Peis, José Verschae, and Andreas Wierz. Primal–dual algorithms for precedence constrained covering problems. *Algorithmica*, 78(3):771–787, 2017. doi:10.1007/S00453-016-0174-3.

32 Julián Mestre. Greedy in approximation algorithms. In *European Symposium on Algorithms*, pages 528–539. Springer, 2006. doi:10.1007/11841036_48.

33 Rolf H. Möhring, Martin Skutella, and Frederik Stork. Scheduling with and/or precedence constraints. *SIAM Journal on Computing*, 33(2):393–415, 2004. doi:10.1137/S009753970037727X.

34 George L Nemhauser and Laurence A Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of Operations Research*, 3(3):177–188, 1978. doi:10.1287/MOOR.3.3.177.

35 George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265–294, 1978. doi:10.1007/BF01588971.

36 James B Orlin. A faster strongly polynomial time algorithm for submodular function minimization. *Mathematical Programming*, 118(2):237–251, 2009. doi:10.1007/S10107-007-0189-2.

37 James G Oxley. *Matroid theory*, volume 3. Oxford University Press, USA, 2006.

38 Jean-Claude Picard. Maximal closure of a graph and applications to combinatorial problems. *Management science*, 22(11):1268–1272, 1976.

39 Thomas Powers, David W Krout, Jeff Bilmes, and Les Atlas. Constrained robust submodular sensor selection with application to multistatic sonar arrays. *IET Radar, Sonar & Navigation*, 11(12):1776–1781, 2017.

40 Richard Rado. Note on independence functions. *Proceedings of the London Mathematical Society*, 3(1):300–320, 1957.

41 Natthawut Samphaiboon and Y Yamada. Heuristic and exact algorithms for the precedence-constrained knapsack problem. *Journal of optimization theory and applications*, 105(3):659–676, 2000.

42 Alexander Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B*, 80(2):346–355, 2000. doi:10.1006/JCTB.2000.1989.

43 Alexander Schrijver et al. *Combinatorial optimization: polyhedra and efficiency*, volume 24 of *Algorithms and Combinatorics*. Springer, 2003.

44 Robert P Streit and Vijay K Garg. The polymatroid representation of a greedoid, and associated galois connections. *arXiv preprint*, 2026. arXiv:2411.15363.

45 Maxim Sviridenko, Jan Vondrák, and Justin Ward. Optimal approximation for submodular and supermodular optimization with bounded curvature. *Mathematics of Operations Research*, 42(4):1197–1218, 2017. doi:10.1287/MOOR.2016.0842.

46 Dávid Szeszlér. Sufficient conditions for the optimality of the greedy algorithm in greedoids. *Journal of Combinatorial Optimization*, 44(1):287–302, 2022. doi:10.1007/S10878-021-00833-Y.

47 William T Tutte. Connectivity in matroids. *Canadian Journal of Mathematics*, 18:1301–1324, 1966.

48 Dominic JA Welsh. *Matroid theory*. Courier Corporation, 2010.

The Code Distortion Problem

Huck Bennett
University of Colorado Boulder, CO, USA

Matthew Fox
University of Colorado Boulder, CO, USA

Bryant Morrell
University of Colorado Boulder, CO, USA

Abstract

Two linear error-correcting codes $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathbb{F}_q^n$ are called *linearly equivalent* if there is a linear isometry mapping $\mathcal{C}_1$ to $\mathcal{C}_2$. In this work, we generalize the notion of linear equivalence and study the minimum distortion $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$ of a linear mapping between codes $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathbb{F}_q^n$, which *quantifies* how similar $\mathcal{C}_1$ and $\mathcal{C}_2$ are. We introduce and study the Code Distortion Problem (CDP), which asks to find a minimum distortion mapping between two input codes $\mathcal{C}_1$ and $\mathcal{C}_2$. CDP generalizes the Linear Code Equivalence Problem (LCE), which is essentially the special case of CDP where $\mathcal{D}(\mathcal{C}_1, C_2) = 1$ and which is well-studied because of its role in cryptography.

We prove that (decisional) CDP is NP-hard to approximate to within any constant factor, and that it is in Σ_2^{P}. We also give a single-exponential-time k^2-approximation algorithm for CDP, where k is the dimension of the input codes. Furthermore, we give a single-exponential-time $\left(\frac{2k+1}{3}\right)^2$-approximation algorithm for a natural special case of CDP, and we show that our analysis is tight in this case.

We use techniques from analogous work on the Lattice Distortion Problem (LDP) by Bennett, Dadush, and Stephens-Davidowitz (ESA, 2016). We also introduce or study a number of additional concepts that might be of independent interest. These include an adaptation of the celebrated reduction of Goldreich, Micciancio, Safra, and Seifert (IPL, 1999) from the Shortest Vector Problem (SVP) to the Closest Vector Problem (CVP) on lattices to the analogous problems on codes; successive minima bases for codes; and the matrix $0 \to 0$ "norm" on subspaces.

2012 ACM Subject Classification Theory of computation → Error-correcting codes; Theory of computation → Random projections and metric embeddings; Theory of computation → Problems, reductions and completeness; Theory of computation → Approximation algorithms analysis

Keywords and phrases Code Distortion, Code Equivalence, Error-Correcting Codes, Metric Embeddings, Approximation Algorithms, Hardness of Approximation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.24

Category APPROX

Funding *Huck Bennett*: Supported by NSF Award CCF-2432132.
Bryant Morrell: Supported by NSF Award CCF-2432132.

Acknowledgements We thank Alexander Golovnev and Noah Stephens-Davidowitz for useful conversations [20], and in particular for allowing us to include Theorem 22 about the coNP-hardness of approximating the matrix $0 \to 0$ norm on subspaces in this work. We also thank the anonymous APPROX reviewers for helpful comments, and in particular for identifying an incorrect inequality (which is now fixed).

1 Introduction

A *linear error-correcting code* (or simply, *code*) is a linear subspace $\mathcal{C} \subseteq \mathbb{F}_q^n$. Although best known for their use in robust communication, codes also appear prominently in algorithms, computational complexity, and cryptography. Notably, several of the cryptosystems appearing


© Huck Bennett, Matthew Fox, and Bryant Morrell;
licensed under Creative Commons License CC-BY 4.0

in late rounds or standardized by the National Institute of Standards and Technology's (NIST) post-quantum cryptography standardization process are based on the intractability of certain computational problems on codes [1, 12, 17].

One central problem on codes of cryptographic interest is the *Linear Code Equivalence Problem* (LCE), in which, given generator matrices of two codes $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathbb{F}_q^n$ as input, the goal is to decide whether $\mathcal{C}_1$ and $\mathcal{C}_2$ are *linearly equivalent*, i.e., whether one can convert $\mathcal{C}_1$ into $\mathcal{C}_2$ simply by permuting coordinates and scaling coordinates by non-zero values. In particular, the LESS cryptosystem [12, 5] crucially relies on the hardness of LCE. It is currently a second-round candidate in NIST's "Additional Digital Signature Schemes" standardization process [26]. Furthermore, a large body of work has studied LCE and its variants from an algorithmic and complexity-theoretic standpoint. See, e.g., [21, 28, 30, 4, 10, 6, 15, 9, 27, 7].

A linear isometry (i.e., distance-preserving map) on $\mathbb{F}_q^n$ corresponds to a matrix $M \in \mathbb{F}_q^{n\times n}$ such that for all $\boldsymbol{x} \in \mathbb{F}_q^n$, $\|M\boldsymbol{x}\|_0 = \|\boldsymbol{x}\|_0$, where $\|\cdot\|_0$ denotes the Hamming weight of a vector. It is not hard to check that a matrix M satisfying this property can be written as the product $M = DP$ of a non-singular diagonal matrix D (which corresponds to scaling) and a permutation matrix P (such matrices M are called *monomial matrices*). In other words, the two operations in the definition of LCE – permuting and scaling coordinates by non-zero values – exactly characterize the linear isometries on $\mathbb{F}_q^n$.

So, LCE asks whether there is a (perfectly) distance-preserving linear map between two input codes $\mathcal{C}_1, \mathcal{C}_2$. However, it is natural to ask a refined version of this question:

Is there a linear map T such that $T(\mathcal{C}_1) = \mathcal{C}_2$ and T *approximately* preserves distances?

We make this precise and quantitative by asking what the minimum *distortion* of such a map T from $\mathcal{C}_1$ to $\mathcal{C}_2$ is. We define the distortion of a map T with $T(\mathcal{C}_1) = \mathcal{C}_2$ as

$$\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) := \Big(\max_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\Big) \Big/ \Big(\min_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\Big) \,, \tag{1}$$

and we define $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) := \min_T \mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2)$, where the minimum is taken over all linear maps T such that $T(\mathcal{C}_1) = \mathcal{C}_2$. Informally, $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2)$ is the ratio of the most T blows up distances and the most it shrinks distances. We note that $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2)$ is simply the distortion of the map T in the standard sense of metric embeddings, where the codes $\mathcal{C}_1$ and $\mathcal{C}_2$ are the ambient spaces and each is equipped with the Hamming metric. (See, e.g., the lecture notes of Matoušek [22] for background on metric embeddings.) Moreover, $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$ is the minimum distortion over all such linear maps T, and it therefore quantifies how similar $\mathcal{C}_1$ and $\mathcal{C}_2$ are. Intuitively, linearly equivalent codes are essentially the same, and codes $\mathcal{C}_1, \mathcal{C}_2$ with $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \gg 1$ are quite different. One can show that $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \geq 1$ (see Lemma 15), and that equality holds if and only if there exist linearly equivalent "unary scalings" of $\mathcal{C}_1$ and $\mathcal{C}_2$ (see Proposition 19).

Recall that a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ of dimension k, called an $[n, k]_q$ code, is often represented by a generator matrix (basis) $G \in \mathbb{F}_q^{n\times k}$.[1] I.e., the code generated by G is $\mathcal{C}(G) := \{G\boldsymbol{x} : \boldsymbol{x} \in \mathbb{F}_q^k\}$. And, one can represent an arbitrary linear map between two $[n, k]_q$ codes in terms of generator matrices $G_1 := (\boldsymbol{g}_1^{(1)}, \ldots, \boldsymbol{g}_k^{(1)}) \in \mathbb{F}_q^{n\times k}$ and $G_2 := (\boldsymbol{g}_1^{(2)}, \ldots, \boldsymbol{g}_k^{(2)}) \in \mathbb{F}_q^{n\times k}$ of the respective codes by $\boldsymbol{g}_i^{(1)} \mapsto \boldsymbol{g}_i^{(2)}$.

[1] In this work we use *column* bases for codes, although row bases are often used in the literature.

For example, consider the generator matrices

$$G_1 := \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{pmatrix} \in \mathbb{F}_2^{4\times 2}\ , \qquad G_2 := \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 1 \\ 0 & 1 \end{pmatrix} \in \mathbb{F}_2^{4\times 2}\ , \qquad G_2' := \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \in \mathbb{F}_2^{4\times 2}\ .$$

Notice that $\mathcal{C}_1 := \mathcal{C}(G_1)$ and $\mathcal{C}_2 := \mathcal{C}(G_2)$ are not linearly equivalent, but that $\mathcal{C}_2 = \mathcal{C}(G_2')$, i.e., G_2 and G_2' are different generator matrices of the same code. Letting T and T' be linear maps such that $T(G_1) = G_2$ and $T'(G_1) = G_2'$, it is straightforward to check that

$$\begin{aligned}\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) &= \Big(\max_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\Big) \Big/ \Big(\min_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\Big) \\ &= \Big(\frac{\|(1,1,1,0)\|_0}{\|(1,0,0,0)\|_0}\Big) \Big/ \Big(\frac{\|(1,0,0,1)\|_0}{\|(1,1,1,1)\|_0}\Big) \\ &= \frac{3}{1} \cdot \Big(\frac{1}{2}\Big)^{-1} = 6\ ,\end{aligned}$$

and

$$\begin{aligned}\mathcal{D}_{T'}(\mathcal{C}_1, \mathcal{C}_2) &= \Big(\max_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}} \frac{\|T'\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\Big) \Big/ \Big(\min_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}} \frac{\|T'\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\Big) \\ &= \Big(\frac{\|(1,0,0,1)\|_0}{\|(1,0,0,0)\|_0}\Big) \Big/ \Big(\frac{\|(1,1,1,0)\|_0}{\|(1,1,1,1)\|_0}\Big) \\ &= \frac{2}{1} \cdot \Big(\frac{3}{4}\Big)^{-1} = \frac{8}{3}\ .\end{aligned}$$

In other words, T' is a substantially lower distortion mapping between $\mathcal{C}_1$ and $\mathcal{C}_2$ than T is. Part of the reason for this is that G_2' is a better basis of $\mathcal{C}_2$ than G_2 is in a sense that we will make precise in the sequel.

In this work, we study code distortion from a computational standpoint. Let $\gamma \geq 1$ be an approximation factor. Specifically, we study the γ-approximate Code Distortion Problem (γ-CDP), which asks to compute a linear mapping T between two codes $\mathcal{C}_1$ and $\mathcal{C}_2$ such that $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) \leq \gamma \cdot \mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$. We also study its decision version, γ-GapCDP, in which the goal is to decide whether $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \leq D$ or $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) > \gamma D$ for a given input distortion threshold value $D \geq 1$. When emphasizing the finite field $\mathbb{F}_q$ that the input codes are over, we add a subscript q to the problems.

1.1 Our Results

We study CDP from both an algorithmic and complexity-theoretic perspective. We first prove hardness of approximation for GapCDP with (fixed) $D > 1$.

▶ **Theorem 1.** *For any constant $\gamma \geq 1$, any constant $D > 1$, and any prime power $q \leq \mathrm{poly}(n)$, γ-GapCDP$_q$ with distortion threshold D is* NP*-hard under deterministic Cook reductions.*

As we formalize in Proposition 19, GapCDP with distortion threshold $D = 1$ is equivalent to LCE up to scaling. From this it follows that GapCDP with $D = 1$ is in NP, but, because of a result of Petrank and Roth [28] showing that LCE is in coAM, it is not NP-hard unless the polynomial hierarchy collapses. In contrast, Theorem 1 asserts that if D is just slightly larger than 1 then GapCDP is NP-hard even to approximate to within any constant factor γ.

For example, taking $D = 1.01$ and $\gamma = 1000$, Theorem 1 shows that it is NP-hard to decide whether $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \leq 1.01$ or $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) > 1010$ when one of these is promised to hold. The result holds for codes $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathbb{F}_q^n$ for any field $\mathbb{F}_q$ with $q \leq \text{poly}(n)$.

We further show in Theorem 24 that GapCDP is in the complexity class Σ_2^{P}. Interestingly, it is not clear that GapCDP is contained in NP. This stands in contrast both to LCE and the Lattice Distortion Problem (LDP), the analogous problem to GapCDP on lattices, which [8] noted is in NP.[2] The main difference between LDP and GapCDP is that the $2 \to 2$ norm $\|T\|$ of a real-valued matrix T is efficient to compute (indeed, $\|T\|$ is simply the largest singular value of T), whereas the problem of computing the $0 \to 0$ "norm" of a linear operator T mapping one code to another is coNP-hard even to approximate to within any constant factor! See Corollary 23.[3]

We next give algorithms for CDP. In Lemma 27, we give and analyze a brute force, exact algorithm for CDP_q on $[n,k]_q$ codes, which runs in $O^*(q^{k(k+1)}) \approx q^{k^2}$ time – essentially the amount of time it takes to enumerate all generator matrices of a given $[n,k]_q$ code.[4] It is unclear how to get a faster exact algorithm, and so we turn to giving approximation algorithms. We first give a single-exponential-time, k^2-approximation algorithm for CDP on $[n,k]_q$ codes.

▶ **Theorem 2.** *For any prime power q, there is an algorithm for k^2-CDP on $[n,k]_q$ codes that runs in $O^*(q^k)$ time and $\text{poly}(n, \log q)$ space.*

We contrast the approximation factor of k^2 in Theorem 2 with the trivial bound bound of $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) \leq n^2$, which holds for any linear map T such that $T(\mathcal{C}_1) = \mathcal{C}_2$ for $[n,k]_q$ codes $\mathcal{C}_1, \mathcal{C}_2$; see Lemma 15. Since $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) \geq 1$ also always holds, this implies a trivial n^2-approximation algorithm for GapCDP. We do not know of a better bound (even a non-constructive one), although we show an upper bound of n when allowing for *non-linear* maps between the codes; see Lemma 17. However, even in this case it is not clear that *finding* such a map would be at all efficient, and we note that Theorem 2 holds for approximate *search* CDP.

For an $[n,k]_q$ code $\mathcal{C}$ and $i \in [k]$, we define the *ith successive minimum* $\lambda_i(\mathcal{C})$ of $\mathcal{C}$ to be the minimum value $r \in \{1, \ldots, n\}$ such that $\mathcal{C}$ contains i linearly independent vectors of Hamming weight at most r. In particular, $\lambda_1(\mathcal{C})$ is the minimum distance of $\mathcal{C}$. We get a strengthening of Theorem 2 when the input codes are binary and when all of the successive minima of both of the input codes are the same.

▶ **Theorem 3.** *There is an algorithm for $\left(\frac{2k+1}{3}\right)^2$-CDP on $[n,k]_2$ codes $\mathcal{C}_1, \mathcal{C}_2$ satisfying $\lambda_1(\mathcal{C}_1) = \cdots = \lambda_k(\mathcal{C}_1) = \lambda_1(\mathcal{C}_2) = \cdots = \lambda_k(\mathcal{C}_2)$ that runs in $O^*(2^k)$ time and $\text{poly}(n)$ space.*

In fact, the *algorithm* used to prove both Theorems 2 and 3 is the same, but all of the codes' successive minima being the same allows for a tighter analysis of its approximation factor. In fact, the analysis of Theorem 3 is tight. However, due to space constraints, we defer the proof to the full version of our paper. Additionally, we note that although we give Theorem 3 only for $\mathbb{F}_2$, one can show that it extends to $\mathbb{F}_q$ for arbitrary prime powers q with a more complicated proof; see [25].

---

[2] Recall that a *lattice* is the analog of a code over the integers or real numbers. Specifically, the lattice generated by a basis $B \in \mathbb{R}^{n \times k}$ with full column rank is $\mathcal{L}(B) := \{B\boldsymbol{z} : \boldsymbol{z} \in \mathbb{Z}^k\}$.

[3] The $0 \to 0$ norm of a linear operator acting on all of $\mathbb{F}_q^n$ (and not restricted restricted to a code/subspace) is also efficiently computable; see Proposition 20.

[4] In this paper, we use $O^*(\cdot)$ to suppress polynomial factors in n and $\log q$.

1.2 Overview of Techniques

At a high level, the proofs of Theorems 1 and 2 follow along similar lines to those used to show upper and lower bounds for the Lattice Distortion Problem (LDP) in [8]. However, they also require a number of additional concepts, which may be of independent interest. The proof of Theorem 3 goes along similar lines to the proof of Theorem 2, but it is substantially more nuanced.

Approximation algorithms

We prove a constructive upper bound using what we call *successive minima bases* for codes. We define a successive minima basis of an $[n,k]_q$ code $\mathcal{C}$ to be a generator matrix $G := (\boldsymbol{g}_1, \ldots, \boldsymbol{g}_k) \in \mathbb{F}_q^{n\times k}$ such that $\|\boldsymbol{g}_i\|_0 = \lambda_i(\mathcal{C})$, i.e., the columns of G are linearly independent and their Hamming weights achieve the successive minima of $\mathcal{C}$. Every code has a successive minima basis (essentially because any k linearly independent vectors in $\mathcal{C}$ form a basis of $\mathcal{C}$) and they can be computed in roughly q^k time; see Lemma 6. In contrast, not every lattice $\mathcal{L}$ has a successive minima basis.[5] Although successive minima are widely used in the study of lattices, to the best of our knowledge they have not previously been explicitly used for codes.

Our approximation algorithm for CDP that leads to Theorems 2 and 3 is simply to compute successive minima bases $G_1 := (\boldsymbol{g}_1^{(1)}, \ldots, \boldsymbol{g}_k^{(1)}) \in \mathbb{F}_q^{n\times k}$ and $G_2 := (\boldsymbol{g}_1^{(2)}, \ldots, \boldsymbol{g}_k^{(2)}) \in \mathbb{F}_q^{n\times k}$ of the respective input $[n,k]_q$ codes $\mathcal{C}_1$ and $\mathcal{C}_2$, and to take the mapping $T : \boldsymbol{g}_i^{(1)} \mapsto \boldsymbol{g}_i^{(2)}$ that they induce.

We show that such a mapping T is a k^2-approximation, proving Theorem 2, as follows. For $[n,k]_q$ codes $\mathcal{C}_1$ and $\mathcal{C}_2$, define $M(\mathcal{C}_1,\mathcal{C}_2) := \max_{i\in[k]} \lambda_i(\mathcal{C}_1)/\lambda_i(\mathcal{C}_2)$. (An analogous quantity for lattices appeared in [8].) We show that the mapping T defined above in terms of successive minima bases satisfies $\mathcal{D}_T(\mathcal{C}_1,\mathcal{C}_2) \le k^2 \cdot M(\mathcal{C}_1,\mathcal{C}_2)M(\mathcal{C}_2,\mathcal{C}_1)$. Furthermore, we show that *any* linear map T' such that $T'\mathcal{C}_1 = \mathcal{C}_2$ must satisfy $\mathcal{D}_{T'}(\mathcal{C}_1,\mathcal{C}_2) \ge M(\mathcal{C}_1,\mathcal{C}_2)M(\mathcal{C}_2,\mathcal{C}_1)$, and therefore $\mathcal{D}(\mathcal{C}_1,\mathcal{C}_2) \ge M(\mathcal{C}_1,\mathcal{C}_2)M(\mathcal{C}_2,\mathcal{C}_1)$. Combining these two bounds, we have that

$$M(\mathcal{C}_1,\mathcal{C}_2)M(\mathcal{C}_2,\mathcal{C}_1) \le \mathcal{D}(\mathcal{C}_1,\mathcal{C}_2) \le \mathcal{D}_T(\mathcal{C}_1,\mathcal{C}_2) \le k^2 \cdot M(\mathcal{C}_1,\mathcal{C}_2)M(\mathcal{C}_2,\mathcal{C}_1)\ ,$$

and therefore that T is k^2-approximately optimal, i.e., that $\mathcal{D}_T(\mathcal{C}_1,\mathcal{C}_2) \le k^2 \cdot \mathcal{D}(\mathcal{C}_1,\mathcal{C}_2)$.

Showing the refined result in Theorem 3 for codes over $\mathbb{F}_2$ all of whose successive minima are the same requires more work. We assume that G_1 and G_2 are successive minima bases of the input $[n,k]_2$ codes (so that each vector in G_1 and G_2 has Hamming weight $\lambda := \lambda_1(\mathcal{C}_1) = \cdots = \lambda_k(\mathcal{C}_1) = \lambda_1(\mathcal{C}_2) = \cdots = \lambda_k(\mathcal{C}_2)$), and we let T be a mapping such that $TG_1 = G_2$. We then analyze

$$\mathcal{D}_T(\mathcal{C}_1,\mathcal{C}_2) := \max_{\boldsymbol{a}\in\mathbb{F}_2^k}\left(\frac{\|G_2\boldsymbol{a}\|_0}{\|G_1\boldsymbol{a}\|_0}\right)\max_{\boldsymbol{b}\in\mathbb{F}_2^k}\left(\frac{\|G_1\boldsymbol{b}\|_0}{\|G_2\boldsymbol{b}\|_0}\right) = \max_{\boldsymbol{a},\boldsymbol{b}\in\mathbb{F}_2^k}\left(\frac{\|G_1\boldsymbol{b}\|_0}{\|G_1\boldsymbol{a}\|_0}\frac{\|G_2\boldsymbol{a}\|_0}{\|G_2\boldsymbol{b}\|_0}\right)$$

[5] As an explicit example, consider the lattice $\mathcal{L} = \mathcal{L}(B) \subset \mathbb{R}^5$ generated by the basis

$$B = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_5) := \begin{pmatrix} 1 & 0 & 0 & 0 & 1/2 \\ 0 & 1 & 0 & 0 & 1/2 \\ 0 & 0 & 1 & 0 & 1/2 \\ 0 & 0 & 0 & 1 & 1/2 \\ 0 & 0 & 0 & 0 & 1/2 \end{pmatrix}\ .$$

One can check that $\boldsymbol{e}_i \in \mathcal{L}$ for $i = 1, \ldots, 5$, and that every vector in $\mathcal{L}$ has norm at least 1. So, $\lambda_1(\mathcal{L}) = \cdots = \lambda_5(\mathcal{L}) = 1$. However, any basis of $\mathcal{L}$ must contain $\pm\boldsymbol{b}_5$, which has norm greater than 1.

We upper bound the quantity $\frac{\|G_2\boldsymbol{a}\|_0\|G_1\boldsymbol{b}\|_0}{\|G_1\boldsymbol{a}\|_0\|G_2\boldsymbol{b}\|_0}$ on the right-hand side for non-zero $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{F}_q^k$ in cases. In each case, we make use of the fact that for non-zero $\boldsymbol{x}$, $\lambda \leq \|G_1\boldsymbol{x}\|_0 \leq \lambda\|\boldsymbol{x}\|_0$, where the upper bound follows from the triangle inequality. Similar bounds hold for G_2.

If $\|\boldsymbol{a}\|_0 + \|\boldsymbol{b}\|_0 \leq \left(\frac{2k+1}{3}\right)$, then by the AM-GM inequality $\|\boldsymbol{a}\|_0\|\boldsymbol{b}\|_0 \leq \left(\frac{2k+1}{3}\right)^2\lambda^2$. So,

$$\frac{\|G_2\boldsymbol{a}\|_0\|G_1\boldsymbol{b}\|_0}{\|G_1\boldsymbol{a}\|_0\|G_2\boldsymbol{b}\|_0} \leq \frac{\left(\frac{2k+1}{3}\right)^2\lambda^2}{\lambda^2} = \left(\frac{2k+1}{3}\right)^2 .$$

On the other hand, if $\|\boldsymbol{a}\|_0 + \|\boldsymbol{b}\|_0 > \left(\frac{2k+1}{3}\right)$ then the supports of $\boldsymbol{a}$ and $\boldsymbol{b}$ have large intersection, which leads to a similar upper bound on $\frac{\|G_2\boldsymbol{a}\|_0\|G_1\boldsymbol{b}\|_0}{\|G_1\boldsymbol{a}\|_0\|G_2\boldsymbol{b}\|_0}$.

Hardness of approximation

To show hardness of approximation for GapCDP, we roughly follow the approach used in [8] for showing hardness of approximation for LDP. We give a pair of reductions, the first of which reduces from the Minimum Distance Problem (GapMDP) to a promise variant of the Nearest Codeword Problem (GapNCP). This first reduction is an adaptation of the reduction from the Shortest Vector Problem (GapSVP) to the Closest Vector Problem (GapCVP) on lattices from [19], and it is likely of independent interest.[6]

The second reduction is from GapNCP to GapCDP. We recall that the input to GapNCP is a pair consisting of a generator matrix $G \in \mathbb{F}_q^{n\times k}$ of a code $\mathcal{C}$ and a target vector $\boldsymbol{t} \in \mathbb{F}_q^n$, and the goal is to decide whether the minimum distance between $\boldsymbol{t}$ and a codeword in $\mathcal{C}$ (i.e., $\min_{\boldsymbol{x}\in\mathcal{C}} \|\boldsymbol{t} - \boldsymbol{x}\|_0$) is at most some threshold. The idea behind our hardness reduction is to construct the following pair of generator matrices, corresponding to an instances of CDP:

$$G_1 := \begin{pmatrix} G & \mathbf{0} \\ 0 & \mathbf{1}_r \end{pmatrix} , \qquad G_2 := \begin{pmatrix} G & -\boldsymbol{t} \\ 0 & \mathbf{1}_r \end{pmatrix} . \tag{2}$$

Here $r \in \mathbb{Z}^+$. Let $\mathcal{C}_1 := \mathcal{C}(G_1)$ and $\mathcal{C}_2 := \mathcal{C}(G_2)$.

If $\min_{\boldsymbol{c}\in\mathcal{C}} \|\boldsymbol{c} - \boldsymbol{t}\|_0$ is small then $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$ is small. If on the other hand $\min_{\boldsymbol{c}\in\mathcal{C}} \|\boldsymbol{c} - \boldsymbol{t}\|_0$ is large, then we would *hope* is that $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$ is large. However, this does not clearly work directly, and we need to modify the reduction. (The attempted reduction sketched here in Equation (2) is the natural analog of the analogous hardness reduction in [8] for LDP; the modified reduction is slightly more involved.)

Because GapMDP is known to be NP-hard to approximate to within any constant [16, 13], this pair of reductions implies that CDP is also NP-hard to approximate to within any constant.

[6] We note that GapMDP and GapNCP are the coding problems analogous to the lattice problems GapSVP and GapCVP, respectively. Furthermore, we note that while [2] showed hardness of approximation for GapNCP, it did not show hardness of the variant that we use in the present work, which has an added promise in the NO case. Finally, we note that our reduction has the advantage of being directly from GapMDP, and of being dimension- and approximation-preserving.

The complexity of GapCDP and its relationship to the $0 \to 0$ norm

We note that the complexity of GapCDP seems closely related to the complexity of computing what we call the matrix $0 \to 0$ *norm restricted to a subspace*. The matrix $0 \to 0$ norm[7]

$$\|T\|_{0\to 0} := \max_{\boldsymbol{x}\in\mathbb{F}_q^n\setminus\{\mathbf{0}\}} \frac{\|M\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} .$$

of a matrix $T \in \mathbb{F}_q^{m\times n}$ is also efficiently computable; see Proposition 20. However, the matrix $0 \to 0$ norm restricted to a subspace $\mathcal{C} \subseteq \mathbb{F}_q^n$ (i.e., code $\mathcal{C}$), defined as,

$$\|T_{|\mathcal{C}}\|_{0\to 0} := \max_{\boldsymbol{x}\in\mathcal{C}\setminus\{\mathbf{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} .$$

is coNP-hard even to approximate; see Corollary 23. We note that for a linear map T with $T\mathcal{C}_1 = \mathcal{C}_2$ for codes $\mathcal{C}_1, \mathcal{C}_2$, $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) = \|T_{|\mathcal{C}_1}\|_{0\to 0} \cdot \|(T^{-1})_{|\mathcal{C}_2}\|_{0\to 0}$. The most obvious approach to showing that GapCDP is in NP would be to use a low-distortion map T between the input codes as a witness, and to have the verifier check that $\|T_{|\mathcal{C}_1}\|_{0\to 0} \cdot \|(T^{-1})_{|\mathcal{C}_2}\|_{0\to 0} \le D$. However, because computing the matrix $0 \to 0$ norm restricted to a subspace is coNP-hard, this approach does not work. (We note that this coNP-hardness does not obviously translate to GapCDP.) As a result, the best upper bound that we get on GapCDP is to show containment in Σ_2^{P}. (As we also note elsewhere, it is a tantalizing question whether one can improve our NP-hardness result for GapCDP to Σ_2^{P}-hardness, and therefore show that GapCDP is Σ_2^{P}-complete.)

1.3 An Approach for a Faster Approximation Algorithm

A natural direction for making our approximation algorithm in Theorem 2 more efficient would be to use reduced (but not necessarily optimal) bases in place of successive minima bases. Indeed, this is how [8] achieves a time-approximation tradeoff in its algorithm for LDP, which includes a polynomial-time approximation algorithm in one regime. (On the other hand, our approximation algorithm in Theorem 2 only runs in polynomial time for an extreme parameter regime – when $k = O(\log_q n)$.) And, a recent line of work has studied basis reduction algorithms for codes [14, 18], including analogs of basis reduction algorithms for lattices. However, it is not clear how to leverage bases reduced in any of the senses they consider to get an approximation algorithm running in a reasonable amount of time and with a reasonable approximation factor.

1.4 Open Questions

Our work leaves open several interesting questions. On the complexity side, our results showing that GapCDP is both NP-hard and in Σ_2^{P} begs the question of whether GapCDP is also Σ_2^{P}-hard, and thus Σ_2^{P}-complete. Moreover, as part of the inspiration for this work comes from the analogous Lattice Distortion Problem (LDP) [8], it is natural to ask whether (approximate) CDP reduces to LDP. We note that a fairly simple reduction from LCE to the

[7] In fact, $\|\cdot\|_{0\to 0}$ is not a matrix norm just as Hamming weight is not a vector norm (even over the real numbers). It does not satisfy the absolute homogeneity property $\|\alpha M\|_{0\to 0} = |\alpha| \, \|M\|_{0\to 0}$ for real-valued matrices M and scalars α, and the magnitude of a finite field element is not even defined. However, the $0 \to 0$ norm satisfies the other axioms of a matrix norm, including the triangle inequality ($\|A + B\|_{0\to 0} \le \|A\|_{0\to 0} + \|B\|_{0\to 0}$), and it is sub-multiplicative (i.e., $\|AB\|_{0\to 0} \le \|A\|_{0\to 0}\|B\|_{0\to 0}$). So, we abuse notation slightly and refer to $\|\cdot\|_{0\to 0}$ as a matrix norm.

Lattice Isomorphism Problem (LIP) – problems that essentially correspond to the distortion $D = 1$ cases of CDP and LDP, respectively – was given in [29, 9]. On the algorithmic side, it is natural to ask whether there is a better exact algorithm than the brute force, roughly q^{k^2}-time algorithm that we give, and whether there is a more efficient approximation algorithm for CDP than our roughly q^k. Finally, we ask given whether it is possible to build cryptography whose security guarantee rests on the hardness of approximate CDP. This seems particularly natural given the role of LCE both as security assumption for the LESS cryptosystem [12], and as a closely related problem to the McEliece cryptosystem [23, 1].[8]

2 Preliminaries

We define a *monomial matrix* M to be the product $M = DP$ of a non-singular diagonal matrix D and a permutation matrix P. We denote the set of $n \times n$ monomial matrices over a field $\mathbb{F}$ as $\mathcal{M}_n(\mathbb{F})$.

2.1 Codes

A *linear code* is a linear subspace $\mathcal{C} \subseteq \mathbb{F}^n$, where $\mathbb{F}$ is field. Generally, $\mathbb{F}$ is the finite field $\mathbb{F}_q$ for some prime power q, and when $q = 2$ (as is common in practice) the code is a *binary code*. A linear code $\mathcal{C}$ is often characterized as an $[n, k]$ or $[n, k, d]_q$ code, where q indicates the order of the field $\mathbb{F}_q$ that the code is taken over, k is the dimension of the code, and d is the minimum Hamming distance of $\mathcal{C}$, i.e.

$$d := \min_{\substack{\boldsymbol{x},\boldsymbol{y}\in\mathcal{C}\\ \boldsymbol{x}\neq\boldsymbol{y}}} \|\boldsymbol{x}-\boldsymbol{y}\|_0 = \min_{\boldsymbol{x}\in\mathcal{C}\setminus\{\boldsymbol{0}\}} \|\boldsymbol{x}\|_0.$$

We also call n the *block length* of the code.

An $[n, k]$ code $\mathcal{C} \subseteq \mathbb{F}^n$ is usually specified by a *generator matrix* $G \in \mathbb{F}^{n\times k}$ with linearly independent columns. The code is then the span of the columns of G, and any *codeword* $\boldsymbol{x} \in \mathcal{C}$ can be expressed as $G\boldsymbol{m}$ for some *message* $\boldsymbol{m} \in \mathbb{F}^k$. Likewise, each message $\boldsymbol{m} \in \mathbb{F}^k$ corresponds to a unique codeword $G\boldsymbol{m} \in \mathcal{C}$.

We will also find it useful to refer to the *support* of a vector $\boldsymbol{x} \in \mathbb{F}^k$. Denoted $\mathrm{supp}(\boldsymbol{x})$, the support of $\boldsymbol{x}$ is the set of indices i for which $\boldsymbol{x}_i \neq 0$. The Hamming weight can then be expressed as $|\mathrm{supp}(\boldsymbol{x})|$, the size of the support.

▶ **Definition 4** (Successive Minima for Codes)**.** *Let $n, k \in \mathbb{Z}^+, k \leq n$, let q be a prime power, and let $\mathcal{C}$ be an $[n, k]_q$ code. For $1 \leq i \leq k$, the* ith successive minimum *of $\mathcal{C}$, denoted $\lambda_i(\mathcal{C})$, is defined as the minimum value $r \in \mathbb{Z}^+$ such that $\mathcal{C}$ contains at least i linearly independent codewords with Hamming weight at most r.*

We note that $\lambda_1(\mathcal{C})$ is the minimum distance d of $\mathcal{C}$. We emphasize again that successive minima are well-studied for lattices, but do not seem to have been studied much for codes. We will make use of the following definition, which formalizes one notion of optimal bases for codes.

▶ **Definition 5** (Successive Minima Basis)**.** *Let $n, k \in \mathbb{Z}^+, k \leq n$, let q be a prime power, and let $\mathcal{C}$ be an $[n, k]_q$ code. A generator matrix $G = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$ of $\mathcal{C}$ is a* successive minima basis *of $\mathcal{C}$ if $\|\boldsymbol{v}_i\|_0 = \lambda_i(\mathcal{C})$ for all $1 \leq i \leq k$.*

[8] One can check that the public and private generator matrices in McEliece generate linearly equivalent codes. However, it is not clear that breaking McEliece reduces to solving LCE or vice-versa.

We next give an algorithm for computing successive minima bases, which in particular implies that such bases always exist.

▶ **Lemma 6.** *Let $n, k \in \mathbb{Z}^+, k \leq n$, let q be a prime power, and let $\mathcal{C}$ be an $[n,k]_q$ code. Then there exists a $O^*(q^k)$-time, polynomial space algorithm for computing a sucessive minima basis of $\mathcal{C}$. In particular, such a basis of $\mathcal{C}$ always exists.*

Proof. Let $\boldsymbol{v}_1 \in \mathcal{C}$ be a codeword that satisfies $\|\boldsymbol{v}_1\| = \lambda_1(\mathcal{C})$, and define $\boldsymbol{v}_i$ recursively for $1 < i \leq k$ so that

$$\boldsymbol{v}_i \in \operatorname*{arg\,min}_{\boldsymbol{x} \in (\mathcal{C} \setminus \operatorname{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{i-1}))} \|\boldsymbol{x}\|_0 \ .$$

We prove inductively that $\|\boldsymbol{v}_i\|_0 = \lambda_i(\mathcal{C})$. This is clearly true in the base case of $i = 1$ since $\boldsymbol{v}_1$ is a shortest non-zero codeword by definition. For the inductive step with $i \geq 2$, assume that we can find $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{i-1} \in \mathcal{C}$ satisfying $\|\boldsymbol{v}_j\|_0 = \lambda_j(\mathcal{C})$ for $1 \leq j \leq i-1$. Additionally, observe that for any $1 \leq i \leq k$, the span of the set $S_i = \{\boldsymbol{v} \in \mathcal{C} : \|\boldsymbol{v}\|_0 \leq \lambda_i(\mathcal{C})\}$ has dimension at least i as a result of there being at least i linearly independent codewords in S_i. On the other hand, the set $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{i-1}\} \subset S_i$ has dimension only $i-1$, so there must exist some $\boldsymbol{v}_i \in \mathcal{C}$ such that $\|\boldsymbol{v}_i\|_0 \leq \lambda_i(\mathcal{C})$ and $\boldsymbol{v}_i \notin \operatorname{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{i-1})$.

We next show that $\|\boldsymbol{v}_i\|_0 \geq \lambda_i(\mathcal{C})$ (and therefore $\|\boldsymbol{v}_i\|_0 = \lambda_i(\mathcal{C})$) for $i \geq 2$. Suppose not. Then $\|\boldsymbol{v}_i\|_0 < \lambda_i(\mathcal{C})$. We have by definition that $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_i$ are linearly independent, and by the induction hypothesis, $\lambda_j(\mathcal{C}) = \|\boldsymbol{v}_j\|_0$ for all $1 \leq j \leq i-1$. By definition, $\|\boldsymbol{v}_1\|_0 \leq \cdots \leq \|\boldsymbol{v}_i\|_0$, and it follows that $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_i \in \mathcal{C}$ are linearly independent vectors all of Hamming weight strictly less than $\lambda_i(\mathcal{C})$, which is a contradiction.

The recursive process above yields k linearly independent codewords $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ with $\|\boldsymbol{v}_i\|_0 = \lambda_i(\mathcal{C})$ for all i. Since $\mathcal{C}$ is a k-dimensional subspace of $\mathbb{F}_q^n$, any k linearly independent vectors in $\mathcal{C}$ (and particularly our $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$) must span $\mathcal{C}$ and thus form a basis for it.

Computing each $\boldsymbol{v}_i$ involves a search over the $O(q^k)$ codewords in $\mathcal{C}$ with polynomial time required to check the linear independence and Hamming weight of each, so it takes $O^*(q^k)$ time to compute. This enumeration can be done in polynomial space. Therefore, by repeating this k times, we find the entire successive minima basis in $O^*(q^k)$ time and polynomial space. ◀

2.2 Coding Problems

Given a code, it is natural to ask what its minimum distance is. The Minimum Distance Problem formalizes this, and we give here the approximation version of the problem.

▶ **Definition 7.** *For $\gamma = \gamma(k) \geq 1$ and a prime power q, the* decisional γ-approximate Minimum Distance Problem over $\mathbb{F}_q$ *(γ-GapMDP$_q$) is the decision problem defined as follows. An instance consists of (a generator matrix $G \in \mathbb{F}_q^{n \times k}$ of) a code $\mathcal{C}$ and an integer distance $0 \leq d \leq m$. It is a:*

- *YES instance if $\lambda_1(\mathcal{C}) \leq d$.*
- *NO instance if $\lambda_1(\mathcal{C}) > \gamma d$.*

We note that GapMDP is NP-hard to approximate to within any constant factor. This was first established by Cheng and Wan [13]. (Earlier work showed NP-hardness under randomized reductions [16], and subsequent work showed simplified deterministic reductions [3, 24].)

▶ **Theorem 8** ([13])**.** *For all constants $\gamma \geq 1$ and all prime powers q, γ-GapMDP$_q$ is* NP*-hard.*

We now define a variant of the Nearest Codeword Problem with a stronger promise in the NO case.

▶ **Definition 9.** *For $\gamma = \gamma(k) \geq 1$, $\alpha = \alpha(k) > 0$, and a prime power q, the* decisional γ-approximate Nearest Codeword Problem over $\mathbb{F}_q$ with distance promise α *(γ-GapNCP$_q^\alpha$) is the decision problem defined as follows. An instance consists of (a generator matrix $G \in \mathbb{F}_q^{n\times k}$ of) a code $\mathcal{C}$, a target vector $\boldsymbol{t} \in \mathbb{F}_q^n$, and a distance parameter $d \in \mathbb{Z}^+$. It is a:*

- *YES instance if* $\operatorname{dist}(\boldsymbol{t}, \mathcal{C}) \leq d$.
- *NO instance if* $\operatorname{dist}(\boldsymbol{t}, \mathcal{C}) > \gamma d$ *and* $d < \alpha\lambda_1(\mathcal{C})$.

We additionally define γ-GapNCP$_q$ to be γ-GapNCP$_q^\infty$. I.e., in γ-GapNCP$_q$ there is no promised upper bound on d in the NO case. We note that γ-GapNCP$_q^\alpha$ trivially reduces to γ-GapNCP$_q^{\alpha'}$ for $\alpha' \geq \alpha$, and in particular γ-GapNCP$_q^\alpha$ trivially reduces to "plain" γ-GapNCP$_q$ for any $\alpha > 0$.

2.3 Matrix Norms

While there are many ways to define matrix norms, the most useful for us is the matrix $p \to q$ norm, where $p, q \in [1, \infty]$. We note in passing that matrix $p \to q$ norms over the real numbers have been studied from a computational standpoint; see, e.g., [11].

We can extend the definition of the matrix $p \to q$ norm to arbitrary (finite) fields $\mathbb{F}$ and to include $p = 0$ and $q = 0$, with $\|\boldsymbol{x}\|_0$ being the Hamming weight as is standard. Note, however, that the matrix $0 \to 0$ "norm" is not in fact a matrix norm as it is not scale-invariant. However, it does satisfy the triangle inequality and it is sub-multiplicative. Ultimately, throughout this article, we abuse notation and nevertheless refer to it as a norm. Additionally, we employ a "restricted" version of the matrix $0 \to 0$ norm, where one only maximizes over those vectors $\boldsymbol{x} \in \mathcal{C}$ for some subspace $\mathcal{C} \subseteq \mathbb{F}^n$. More formally:

▶ **Definition 10.** *Let $n \in \mathbb{Z}^+$, let $\mathbb{F}$ be a field, let $T \in \mathrm{GL}_n(\mathbb{F})$ be a matrix, and let $\mathcal{C} \subseteq \mathbb{F}^n$ be a subspace. The* matrix $0 \to 0$ norm of T restricted to the subspace $\mathcal{C}$ *is the quantity*

$$\|T|_{\mathcal{C}}\|_{0\to 0} \coloneqq \max_{\boldsymbol{x}\in\mathcal{C}\setminus\{\boldsymbol{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}.$$

Note that the restricted matrix $0 \to 0$ norm is also sub-multiplicative.

2.4 Distortion

To compare pairs of general codes, we introduce a quantity that captures how similar they are. We formalize this with *distortion*, similar to the formulation for the Lattice Distortion Problem [8] and distortion on metric embeddings. We start by defining the distortion of a particular transformation between two codes.

▶ **Definition 11.** *Let $n, k \in \mathbb{Z}^+$, $k \leq n$, let q be a prime power, let $\mathcal{C}_1, \mathcal{C}_2$ be $[n, k]_q$ codes, and let T be a linear map such that $T(\mathcal{C}_1) = \mathcal{C}_2$. The* distortion $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2)$ of T *the minimum value D such that for all $\boldsymbol{x} \in \mathcal{C}_1$,*

$$D_1 \cdot \|\boldsymbol{x}\|_0 \leq \|T\boldsymbol{x}\|_0 \leq D_2 \cdot \|\boldsymbol{x}\|_0$$

for some values $D_1, D_2 > 0$ satisfying $D_2/D_1 \leq D$. Equivalently,

$$\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) := \Big(\max_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\boldsymbol{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\Big) \Big/ \Big(\min_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\boldsymbol{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\Big)\ ,$$

which is the definition we gave in Section 1. Moreover, when $T \in \mathrm{GL}_n(\mathbb{F}_q)$ *and thus is invertible, this is equivalent to*

$$\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) = \|T|_{\mathcal{C}_1}\|_{0\to 0} \cdot \|T^{-1}|_{\mathcal{C}_2}\|_{0\to 0}. \tag{3}$$

Consequently, at least in the case when T is invertible, the distortion between two codes $\mathcal{C}_1$ and $\mathcal{C}_2$ is inextricably tied to computing two matrix $0 \to 0$ norms, where one is restricted to $\mathcal{C}_1$ and the other is restricted to $\mathcal{C}_2$. Interestingly, as we show in Lemma 18, the assumption that T is invertible can be made without loss of generality, so we may take Equation (3) as an equivalent definition of distortion. Furthermore, while we require the codes in Definition 11 to have the same block length n, the definition can be extended to codes with different block lengths by simply padding the shorter code with zeros (or, one could allow for non-square matrices T).

We can now define the distortion between two codes generically as the lowest possible distortion between them using any transformation.

▶ **Definition 12.** *Let* $n, k \in \mathbb{Z}^+$, $k \le n$ *and let* q *be a prime power. The* distortion $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$ *between* $[n,k]_q$ *codes* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *is defined as the minimum value* D *such that there exists* $T \in \mathrm{GL}_n(\mathbb{F}_q)$ *with* $T(\mathcal{C}_1) = \mathcal{C}_2$ *and* $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) = D$*. In other words,*

$$\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \coloneqq \min_{\substack{T \in \mathrm{GL}_n(\mathbb{F}_q),\\ T(\mathcal{C}_1)=\mathcal{C}_2}} \mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2)\ . \tag{4}$$

2.4.1 The Code Distortion Problem

Using this definition of code distortion, we now define both the search and decision versions of the (γ-approximate) Code Distortion Problem (CDP).

▶ **Definition 13.** *For* $\gamma = \gamma(n) \ge 1$ *and a prime power* q*, the* γ-approximate Code Distortion Problem over $\mathbb{F}_q$ *(γ-*CDP_q*) is the search problem defined as follows. An instance consists of (generator matrices* $G_1, G_2 \in \mathbb{F}_q^{m\times n}$ *of) codes* $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathbb{F}_q^n$*, and the goal is to output* D *such that* $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \le D \le \gamma \cdot \mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$*.*

▶ **Definition 14.** *For* $\gamma = \gamma(n) \ge 1$ *and a prime power* q*, the* decisional γ-approximate Code Distortion Problem over $\mathbb{F}_q$ *(γ-*GapCDP_q*) is the decision problem defined as follows. An instance consists of (generator matrices* $G_1, G_2 \in \mathbb{F}_q^{m\times n}$ *of) codes* $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathbb{F}_q^n$ *and* $D \ge 1$*. It is a:*

- *YES instance if* $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \le D$*.*
- *NO instance if* $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) > \gamma D$*.*

The exact versions of these problems, CDP_q and GapCDP_q, are then defined to be 1-CDP_q and 1-GapCDP_q, respectively. Furthermore, the Linear Code Equivalence Problem is essentially the special case of CDP with $D = 1$. We formalize this relationship in Proposition 19 in the next section.

2.5 Basic Facts about Distortion

We next present several basic facts about the distortion between codes. We start with the following bounds on $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$.

▶ **Lemma 15.** *Let* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *be* $[n,k]_q$ *codes. Then, for any linear map* T *such that* $T(\mathcal{C}_1) = \mathcal{C}_2$*,* $1 \le \mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) \le n^2$*. As a consequence,* $1 \le \mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \le n^2$

Proof. For the lower bound, simply note that by the sub-multiplicativity of the matrix $0 \to 0$ norm on subspaces, it holds for all $T \in \mathrm{GL}_n(\mathbb{F})$ such that $T(\mathcal{C}_1) = \mathcal{C}_2$ that

$$1 = \|(T^{-1}T)|_{\mathcal{C}_1}\|_{0\to 0} \leq \|T^{-1}|_{\mathcal{C}_2}\|_{0\to 0} \cdot \|T|_{\mathcal{C}_1}\|_{0\to 0} = \mathcal{D}_T(C_1, \mathcal{C}_2).$$

For the upper bound, note that for all $T \in \mathrm{GL}_n(\mathbb{F})$ such that $T(\mathcal{C}_1) = \mathcal{C}_2$, it holds that

$$\|T|_{\mathcal{C}_1}\|_{0\to 0} = \max_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} \leq \max_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}} \|T\boldsymbol{x}\|_0 \leq n \ ,$$

where the last inequality follows from the fact that $\mathcal{C}_2$ is an $[n,k]_q$ code and for all $x \in \mathcal{C}_1\setminus\{\mathbf{0}\}$, $T\boldsymbol{x} \in \mathcal{C}_2\setminus\{\mathbf{0}\}$. A similar argument establishes that $\|T^{-1}|_{\mathcal{C}_2}\|_{0\to 0} \leq n$, and consequently

$$\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) = \|T|_{\mathcal{C}_1}\|_{0\to 0} \cdot \|T^{-1}|_{\mathcal{C}_2}\|_{0\to 0} \leq n^2 \ ,$$

as desired. The fact that $1 \leq \mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \leq n^2$ then follows from the definition of $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$. ◀

It is an open question if the upper bound in Lemma 15 can be reduced below n^2. Indeed, we do not know an example of a code with distortion n^2, but we do know an example of a code with distortion n, which we detail in the following.

▶ **Lemma 16.** *For every $n \in \mathbb{Z}^+$, there exist $[n,2]_2$ codes $\mathcal{C}_1, \mathcal{C}_2$ such that $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) = n$.*

Proof. Let $\mathcal{C}_1, \mathcal{C}_2$ be the codes defined as follows:

$$\mathcal{C}_1 = \left\{\mathbf{0}_n, \begin{pmatrix}1\\ \mathbf{0}_{n-1}\end{pmatrix}, \begin{pmatrix}0\\ \mathbf{1}_{n-1}\end{pmatrix}, \mathbf{1}_n\right\}, \quad \mathcal{C}_2 = \left\{\mathbf{0}_n, \begin{pmatrix}1\\1\\0\\ \mathbf{0}_{n-3}\end{pmatrix}, \begin{pmatrix}0\\1\\1\\ \mathbf{0}_{n-3}\end{pmatrix}, \begin{pmatrix}1\\0\\1\\ \mathbf{0}_{n-3}\end{pmatrix}\right\}.$$

It is easy to check that $\mathcal{C}_1$ and $\mathcal{C}_2$ are linear codes. Each non-zero codeword in $\mathcal{C}_2$ has Hamming weight 2 (the constant weight of non-zero codewords in $\mathcal{C}_2$ is the only property of it that we are using) while the non-zero codewords in $\mathcal{C}_1$ have minimum weight 1 and maximum weight n. As such, the distortion for any transformation T from $\mathcal{C}_1$ to $\mathcal{C}_2$ is

$$\max_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} \max_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}} \frac{\|\boldsymbol{x}\|_0}{\|T\boldsymbol{x}\|_0} = \frac{2}{1} \cdot \frac{n}{2} = n \ .$$ ◀

Ultimately, while we are unable to prove a matching upper bound of n on $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$, we are nevertheless able to prove such a bound when including *non-linear transformations* T.

▶ **Lemma 17.** *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be $[n,k]_q$ codes. Then, $\min_T \mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) \leq n$, where the minimum is over all maps T (including non-linear maps) such that $T(\mathcal{C}_1) = \mathcal{C}_2$.*

Proof. List all non-zero codewords in $\mathcal{C}_1$ in monotonically increasing order of Hamming weight, and prepare a similar list for $\mathcal{C}_2$. Let T be the map that takes codeword i in the first list to codeword i in the second list. Let $N := q^k - 1$, and let $1 \leq a_1 \leq \cdots \leq a_N \leq n$ and $1 \leq b_1 \leq \cdots \leq b_N \leq n$ be the Hamming weights of the non-zero codewords in the respective lists. Then, the distortion of T is

$$\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) = \left(\max_{i\in[N]} \frac{a_i}{b_i}\right) \cdot \left(\max_{j\in[N]} \frac{b_j}{a_j}\right) \ .$$

Now put $i^* = \arg\max_{i\in[N]} a_i/b_i$, and let $M = a_{i^*}/b_{i^*}$. We assume without loss of generality that $M \geq 1$ (otherwise, repeat the argument with the two codes flipped). Then, for all $j > i^*$,

$b_j/a_j \le n/a_{i^*} \le nb_{i^*}/a_{i^*} = n/M$. And, for all $j \le i^*$, $b_j/a_j \le b_{i^*}/1 = a_{i^*}/M \le n/M$. Consequently, for all $j \in [N]$, $b_j/a_j \le n/M$. Therefore,

$$\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) = \left(\max_{i\in[N]} \frac{a_i}{b_i}\right) \cdot \left(\max_{j\in[N]} \frac{b_j}{a_j}\right) \le M \cdot \frac{n}{M} = n \ ,$$

as desired. ◀

We next show that if there exists a matrix T such that $T\mathcal{C}_1 = \mathcal{C}_2$, then then exists an efficiently computable, *invertible* matrix T' such that $T'\mathcal{C}_1 = \mathcal{C}_2$ and $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) = \mathcal{D}_{T'}(\mathcal{C}_1, \mathcal{C}_2)$.

▶ **Lemma 18.** *Let T be a (not necessarily invertible) matrix such that $T(\mathcal{C}_1) = \mathcal{C}_2$ for $[n,k]_q$ codes $\mathcal{C}_1$, $\mathcal{C}_2$ such that $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) \le D$. Then there exists an invertible matrix T' such that $T'(\mathcal{C}_1) = \mathcal{C}_2$ and $\mathcal{D}_{T'}(\mathcal{C}_1, \mathcal{C}_2)$.*

Proof. Now consider a generator matrix $G_1 \in \mathbb{F}_q^{n\times k}$ of $\mathcal{C}_1$ and the corresponding generator matrix of $\mathcal{C}_2$, $G_2 := T(G_1) \in \mathbb{F}_q^{n\times k}$. Now, let $H_1, H_2 \in \mathbb{F}_q^{n\times(n-k)}$ be such that $A := (G_1|H_1), B := (G_2|H_2) \in \mathbb{F}_q^{n\times n}$ are full-rank. (In particular, it suffices to take H_1 and H_2 to be (transposed) parity-check matrices for $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively.) Finally, let $T' := BA^{-1}$. Note that T' is invertible, and that $T\boldsymbol{c} = T'\boldsymbol{c}$ for every $\boldsymbol{c} \in \mathcal{C}$. It follows that $T'(\mathcal{C}_1) = \mathcal{C}_2$ and $\mathcal{D}_{T'}(\mathcal{C}_1, \mathcal{C}_2) = \mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) = D$, as needed. ◀

We next show that two codes $\mathcal{C}_1$ and $\mathcal{C}_2$ being linearly equivalent is equivalent to $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) = 1$ up to "scaling."

▶ **Proposition 19.** *Let $\mathcal{C}_1, \mathcal{C}_2$ be $[n,k]_q$ codes for some $n, k \in \mathbb{Z}^+$ and prime power q. Then $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) = 1$ if and only if $\mathcal{C}_1 \otimes \mathbf{1}_{r_1}$ (padded with some number of zeros to reach block length $r_2 n$) is linearly equivalent to $\mathcal{C}_2 \otimes \mathbf{1}_{r_2}$ for some $r_1, r_2 \in \mathbb{Z}^+$, $r_1 \le r_2$.*

Proof. We start with the backward direction. Let $\mathcal{C}_1'$ be the result of padding $\mathcal{C}_1 \otimes \mathbf{1}_{r_1}$ with $r_2 n - r_1 n$ zeros and let $\mathcal{C}_2' = \mathcal{C}_2 \otimes \mathbf{1}_{r_2}$. Suppose we have $r_1, r_2 \in \mathbb{Z}^+$, $r_1 \le r_2$ such that $\mathcal{C}_1'$ is linearly equivalent to $\mathcal{C}_2'$. Then, there must exist a monomial matrix T such that $T(\mathcal{C}_1') = \mathcal{C}_2'$ and $\|T(\boldsymbol{x} \otimes \mathbf{1}_{r_1})\|_0 = \|\boldsymbol{x} \otimes \mathbf{1}_{r_1}\|_0$ for all $\boldsymbol{x} \in \mathcal{C}_1$. Additionally, we can define $S_1 = I_n \otimes \mathbf{1}_{r_1}$, which maps $\mathcal{C}_1$ to $\mathcal{C}_1'$ with uniform scaling $\|S_1\boldsymbol{x}\|_0 = r_1\|\boldsymbol{x}\|_0$ for any $\boldsymbol{x} \in \mathcal{C}_1$, and $S_2 = I_n \otimes \boldsymbol{e}_1^\top$,[9] which maps $\mathcal{C}_2'$ to $\mathcal{C}_2$ with uniform scaling of $\|S_2\boldsymbol{x}\|_0 = \frac{1}{r_2}\|\boldsymbol{x}\|_0$ for any $\boldsymbol{x} \in \mathcal{C}_2'$. We now compose these to define a transformation $T' = S_2 T S_1$ that maps $\mathcal{C}_1$ to $\mathcal{C}_2$ and note that for any $\boldsymbol{x} \in \mathcal{C}_1$,

$$\frac{\|T'\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} = \frac{\|S_2 T S_1 \boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} = \frac{1}{r_2} \cdot \frac{\|TS_1\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} = \frac{1}{r_2} \cdot \frac{\|S_1\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} = \frac{r_1}{r_2} \cdot \frac{\|\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} = \frac{r_1}{r_2}.$$

As this is true for all $\boldsymbol{x} \in \mathcal{C}_1$, we can see that $\|T'|_{\mathcal{C}_1}\|_{0\to 0} = \frac{r_1}{r_2}$ and $\|(T')^{-1}|_{\mathcal{C}_2}\|_{0\to 0} = \frac{r_2}{r_1}$. Thus, $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) = \mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) = 1$.

For the forward direction, we are now given that $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) = 1$, i.e., there exists some T such that $T(\mathcal{C}_1) = \mathcal{C}_2$ and $\|T\boldsymbol{x}\|_0 = c\|\boldsymbol{x}\|_0$ for all $\boldsymbol{x} \in \mathcal{C}_1$ and some $c \in \mathbb{Q}^+$ (we assume without loss of generality that $c \le 1$). Let $c = \frac{r_1}{r_2}$ for some $r_1, r_2 \in \mathbb{Z}^+$, and once again take $\mathcal{C}_1' = \mathcal{C}_1 \otimes \mathbf{1}_{r_1}$ (padded with $r_2 n - r_1 n$ zeros) and $\mathcal{C}_2' = \mathcal{C}_2 \otimes \mathbf{1}_{r_2}$. Finally, we redefine $S_1 = \left(I_n \otimes \begin{pmatrix}1 & \mathbf{0}_{r_1-1}\end{pmatrix}\right)$, $S_2 = I_n \otimes \mathbf{1}_{r_2}$. As before, note that $S_1(\mathcal{C}_1') = \mathcal{C}_1$, $S_2(\mathcal{C}_2) = \mathcal{C}_2'$, $\|S_1\boldsymbol{x}\|_0 = \frac{1}{r_1}\|\boldsymbol{x}\|_0$ for all $\boldsymbol{x} \in \mathcal{C}_1'$, and $\|S_2\boldsymbol{x}\|_0 = r_2\|\boldsymbol{x}\|_0$ for all $\boldsymbol{x} \in \mathcal{C}_2$. We define $T' = S_2 T S_1$ so that T' maps $\mathcal{C}_1'$ to $\mathcal{C}_2'$, and again compute that

$$\|T'\boldsymbol{x}\|_0 = \|S_2 T S_1 \boldsymbol{x}\|_0 = r_2\|TS_1\boldsymbol{x}\|_0 = cr_2\|S_1\boldsymbol{x}\|_0 = \frac{cr_2}{r_1}\|\boldsymbol{x}\|_0 = \|\boldsymbol{x}\|_0 \ .$$

This is true for all $\boldsymbol{x} \in \mathcal{C}_1'$, so $\mathcal{C}_1' = \mathcal{C}_1 \otimes \mathbf{1}_{r_1}$ is equivalent to $\mathcal{C}_2' = \mathcal{C}_2 \otimes \mathbf{1}_{r_2}$ as required to complete the proof. ◀

[9] Here $\boldsymbol{e}_1 \in \mathbb{F}_q^{r_2}$ is the first standard normal basis vector in r_2 dimensions.

2.6 Reductions

In this paper, we employ three different types of reductions. The first and most general notion is that of a *Turing reduction*. Formally, there is a Turing reduction from a decision problem A to a decision problem B if and only if there exists an algorithm for A, given oracle access to B. In other words, A Turing-reduces to B if, given a subroutine for B, one can decide A. The second type of reduction is a *Cook reduction*. Formally, a Cook reduction is a polynomial-time Turing reduction, i.e., A Cook-reduces to B if and only if A Turing-reduces to B and the overall algorithm runs in polynomial time. The third and final type of reduction is that of a *Karp reduction* (a.k.a., a *polynomial-time many-one reduction*). Formally, a Karp reduction is a deterministic and polynomial-time algorithm that maps YES and NO instances of A to YES and NO instances of B, respectively.

3 The Complexity of the Code Distortion Problem

We now study some basic facts about the complexity of both the exact and approximate versions of GapCDP. Given the relationship of GapCDP to the matrix $0 \to 0$ norm (Section 2.3), we start by discussing the complexity of computing both the unrestricted and restricted versions of the matrix $0 \to 0$ norm.

3.1 The Complexity of the Matrix $0 \to 0$ Norm

Interestingly, the (unrestricted) matrix $0 \to 0$ norm is easy to compute.

▶ **Proposition 20.** *Let $\mathbb{F}$ be a field, let $m, n \in \mathbb{Z}^+$, and let $T = (\boldsymbol{t}_1, \ldots, \boldsymbol{t}_n) \in \mathbb{F}^{m \times n}$. Then,*

$$\|T\|_{0\to 0} = \max_{i\in[n]} \|\boldsymbol{t}_i\|_0 \ .$$

Proof. By the triangle inequality, it holds for all $\boldsymbol{x} \in \mathbb{F}^n \setminus \{\boldsymbol{0}\}$ that

$$\begin{aligned}
\frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} &= \frac{\|\sum_{i\in\mathrm{supp}(\boldsymbol{x})} x_i \boldsymbol{t}_i\|_0}{|\mathrm{supp}(\boldsymbol{x})|} \\
&\leq \frac{\sum_{i\in\mathrm{supp}(\boldsymbol{x})} \|x_i \boldsymbol{t}_i\|_0}{|\mathrm{supp}(\boldsymbol{x})|} \\
&\leq \frac{|\mathrm{supp}(\boldsymbol{x})| \cdot \max_{i\in\mathrm{supp}(\boldsymbol{x})} \|\boldsymbol{t}_i\|_0}{|\mathrm{supp}(\boldsymbol{x})|} \\
&= \max_{i\in\mathrm{supp}(\boldsymbol{x})} \|\boldsymbol{t}_i\|_0 \ .
\end{aligned}$$

Furthermore, $\max_{i\in\mathrm{supp}(x)} \|\boldsymbol{t}_i\|_0 \leq \max_{i\in[n]} \|\boldsymbol{t}_i\|_0$, so $\|T\|_{0\to 0} \leq \max_{i\in[n]} \|\boldsymbol{t}_i\|_0$. On the other hand,

$$\|T\|_{0\to 0} \geq \max_{i\in[n]} \frac{\|T\boldsymbol{e}_i\|_0}{\|\boldsymbol{e}_i\|_0} = \max_{i\in[n]} \|\boldsymbol{t}_i\|_0 \ . \qquad \blacktriangleleft$$

We note that an immediate consequence of Proposition 20 is that there is a $\mathrm{poly}(m, n)$-time algorithm for computing $\|T\|_{0\to 0}$ for a matrix $T \in \mathbb{F}^{m\times n}$: simply compute the maximum Hamming weight of a column of T. That said, we show that matrix $0 \to 0$ norm *restricted to a subspace* is hard to compute, even approximately. We formalize this by defining a decisional version of the problem.

▶ **Definition 21.** *Let $\mathbb{F}$ be a field, let $m, n, k, q \in \mathbb{Z}^+$, and let $\gamma = \gamma(m,n) \geq 1$. The* decisional γ-approximate matrix $0 \to 0$ norm problem over a subspace of $\mathbb{F}_q$ *(γ-GapNorm$_q^{0\to 0}$) is the promise problem defined as follows. An instance consists of a basis $G \subseteq \mathbb{F}_q^{n\times k}$ of a subspace $\mathcal{C} \subseteq \mathbb{F}_q^n$, a matrix $T \in \mathbb{F}_q^{m\times n}$, and threshold $r \geq 0$. It is a:*

- *YES instance if $\|T|_{\mathcal{C}}\|_{0\to 0} \leq r$.*
- *NO instance if $\|T|_{\mathcal{C}}\|_{0\to 0} > \gamma r$.*

We will now give a reduction from γ'-GapMDP$_2$ to γ-GapNorm$_2^{0\to 0}$ for which it suffices to take $\gamma' = O(\gamma)$. This result is due to Golovnev and Stephens-Davidowitz [20].

▶ **Theorem 22** ([20])**.** *Let $\gamma \geq 1$ and let $\epsilon > 0$ be a constant. Then, for all $\gamma' > \left(\frac{1+\epsilon}{1-\epsilon}\right) \cdot \gamma$, there is a Karp reduction from γ'-GapMDP$_2$ to γ-GapNorm$_2^{0\to 0}$.*

Proof. Let $(G \in \mathbb{F}_2^{m\times n}, r')$ be an instance of γ'-GapMDP$_2$, where $m, n \in \mathbb{Z}^+$. Moreover, let T be a generator matrix of an $[m,n]_2$ code that is ε-balanced, i.e., all non-zero codewords in $\mathcal{C}(T)$ have Hamming weight in the interval $[(1-\varepsilon)\frac{m}{2}, (1+\varepsilon)\frac{m}{2}]$. By a result of Ta-Shma [31], such a T exists and, moreover, can be constructed in deterministic polynomial time.

Now, if (G, r') is a YES instance of γ'-MDP$_2$, then for all $\boldsymbol{x} \in \mathcal{C}\backslash\{\mathbf{0}\}$ with $\|\boldsymbol{x}\|_0 = \lambda_1(\mathcal{C})$,

$$\|T|_{\mathcal{C}}\|_{0\to 0} \geq \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} \geq \frac{\|T\boldsymbol{x}\|_0}{r'} \geq \frac{(1-\epsilon)m/2}{r'}.$$

Here, the first inequality uses the definition of the restricted matrix $0 \to 0$ norm, the second uses the fact that (G, r') is a YES instance, and the third uses the fact that T generates an ε-balanced code.

On the other hand, if (G, r') is a NO instance of γ'-MDP$_2$, then for all $\boldsymbol{x} \in \mathcal{C}\backslash\{\mathbf{0}\}$,

$$\frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} < \frac{(1+\epsilon)m/2}{\gamma' r'},$$

which implies

$$\|T|_{\mathcal{C}}\|_{0\to 0} = \max_{\boldsymbol{x}\in\mathcal{C}\backslash\{\mathbf{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} < \frac{(1+\epsilon)m/2}{\gamma' r'}.$$

Altogether, with $\gamma = \frac{1-\varepsilon}{1+\varepsilon}(\gamma' + \delta)$ and $r = \frac{(1+\epsilon)m/2}{(\gamma'+\delta)r'}$ for any $\delta > 0$, it holds that $\|T|_{\mathcal{C}}\|_{0\to 0} > \gamma r$ (a YES instance of co-γ-GapNorm$_2^{0\to 0}$) implies a YES instance of γ'-MDP$_2$, and $\|T|_{\mathcal{C}}\|_{0\to 0} \leq r$ (a NO instance of co-γ-GapNorm$_2^{0\to 0}$) implies a NO instance of γ'-MDP$_2$. ◀

Consequently, by Theorem 8, γ-GapNorm$_2^{0\to 0}$ is coNP-hard. In fact, it is coNP-complete.

▶ **Corollary 23.** *For all $\gamma \geq 1$, γ-GapNorm$_2^{0\to 0}$ is coNP-complete.*

Proof. Since γ-GapNorm$_2^{0\to 0}$ is coNP-hard, it suffices to prove that γ-GapNorm$_2^{0\to 0} \in$ coNP. But this is plain, since if (G, T, r) is a YES instance, then for all $\boldsymbol{x} \in \mathcal{C}\backslash\{\mathbf{0}\}$, $\|T\boldsymbol{x}\|_0/\|\boldsymbol{x}\|_0 \leq r$, and if (G, T, r) is a NO instance, then there exists $\boldsymbol{x} \in \mathcal{C}\backslash\{\mathbf{0}\}$ for which $\|T\boldsymbol{x}\|_0/\|\boldsymbol{x}\|_0 \geq \gamma r$. ◀

3.2 GapCDP is in Σ_2^{P}

We now show that the code distortion problem is in Σ_2^{P}, the second level of the polynomial hierarchy. Recall that a language $L \in \Sigma_2^{\mathsf{P}}$ if and only if there exists two polynomials p_1 and p_2 as well as a polynomial-time deterministic Turing machine M such that for all inputs $x \in \{0,1\}^*$, $x \in L$ if and only if $(\exists y \in \{0,1\}^{p_1(|x|)})(\forall z \in \{0,1\}^{p_2(|x|)}) : M(x,y,z) = 1$.

▶ **Theorem 24.** *For all $q \in \mathbb{Z}^+$, $\mathrm{GapCDP}_q \in \Sigma_2^{\mathsf{P}}$.*

Proof. Let (G_1, G_2, D) be an instance of GapCDP_q, where G_1 and G_2 generate the codes $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively. By definition, (G_1, G_2, D) is a YES instance if and only if $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \leq D$. This holds if and only if there exists $T \in \mathrm{GL}_m(\mathbb{F}_q)$ with $T(\mathcal{C}_1) = \mathcal{C}_2$ such that for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{C}_1 \backslash \{\mathbf{0}\}$,

$$\left(\frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\right) \cdot \left(\frac{\|T\boldsymbol{y}\|_0}{\|\boldsymbol{y}\|_0}\right)^{-1} \leq D.$$

For all such T, $\boldsymbol{x}$, and $\boldsymbol{y}$, the above inequality is checkable in deterministic polynomial time. Moreover, the size of all such T, $\boldsymbol{x}$, and $\boldsymbol{y}$ is a polynomial in the size of the input (G_1, G_2, D). Consequently, as the order of the $\exists$ and $\forall$ quantifiers in the above reformulation of GapCDP_q is consistent with the class Σ_2^{P}, it holds that $\mathrm{GapCDP}_q \in \Sigma_2^{\mathsf{P}}$, as desired. ◀

As mentioned in Section 1.4, we suspect GapCDP_q is also Σ_2^{P}-hard, thus making it Σ_2^{P}-complete. However, proving this remains an interesting open question.

3.3 GMSS for Codes

We will now give a reduction from γ-GapMDP to γ-GapNCP that is analogous to the seminal reduction from the Shortest Vector Problem to the Closest Vector Problem on lattices due to [19]. We will use this result in our proof of NP-hardness of approximate CDP.

▶ **Theorem 25** (GMSS for Codes)**.** *Let $m, n \in \mathbb{Z}^+$ with $n \leq m$, let $\gamma = \gamma(m, n) \geq 1$, and let q be a prime power. Then there is a $\mathrm{poly}(m, q)$-time Turing reduction from γ-GapMDP_q on $[m, n]_q$ codes to γ-$\mathrm{GapNCP}_q^{1/\gamma}$ (and hence also γ-GapNCP_q) on $[m, n-1]_q$ codes.*

Proof. Let $(G = (\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n) \in \mathbb{F}_q^{m \times n}, d)$ be the input instance of γ-GapMDP_q. The reduction does the following. It constructs $(q-1)n$ many instances $(G_i, \boldsymbol{t}_{i,j} := -j \cdot \boldsymbol{g}_i, d)$ of GapNCP_q for $1 \leq i \leq n$ and $j \in \mathbb{F}_q^*$, where $G_i := (\boldsymbol{g}_1, \ldots, \boldsymbol{g}_{i-1}, \boldsymbol{g}_{i+1}, \ldots, \boldsymbol{g}_n)$ is "G with its ith column removed." It then calls its γ-$\mathrm{GapNCP}_q^{1/\gamma}$ oracle on each instance $(G_i, \boldsymbol{t}_{i,j}, d)$. The reduction outputs YES if the oracle responds with YES on some input, and otherwise it outputs NO.

It is clear that the reduction runs in the stated amount of time, and it remains to show its correctness. Suppose that the input is a YES instance. Then there exists $\boldsymbol{a} \in \mathbb{F}_q^n \backslash \{\mathbf{0}\}$ such that $\|G\boldsymbol{a}\|_0 \leq d$. Let a_i be a non-zero coordinate of $\boldsymbol{a}$, and let $\boldsymbol{a}' := (a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n) \in \mathbb{F}_q^{n-1}$ be "$\boldsymbol{a}$ with its ith coordinate deleted." Then for $j = a_i$, we have that

$$\|G_i\boldsymbol{a}' - \boldsymbol{t}_{i,j}\|_0 = \|G_i\boldsymbol{a}' + j\boldsymbol{g}_i\|_0 = \Big\| \sum_{\ell \neq i} a_\ell \boldsymbol{g}_\ell + a_i \boldsymbol{g}_i \Big\|_0 = \|G\boldsymbol{a}\|_0 \leq d \ .$$

So, the γ-$\mathrm{GapNCP}_q^{1/\gamma}$ oracle outputs YES on input $(G_i, \boldsymbol{t}_{i,j}, d)$, as needed.

Now, suppose that the input is a NO instance. Let $\mathcal{C} := \mathcal{C}(G)$ and let $\mathcal{C}_i := \mathcal{C}(G_i)$. We note that for every $1 \leq i \leq n$ and $j \in \mathbb{F}_q^*$,

$$\mathcal{C}_i \subset \mathcal{C} \text{ and } \mathcal{C}_i - \boldsymbol{t}_{i,j} \subset \mathcal{C} \setminus \{\mathbf{0}\} \ .$$

It follows that $\lambda_1(\mathcal{C}_i) \geq \lambda_1(\mathcal{C})$ and that $\mathrm{dist}(\boldsymbol{t}_{i,j}, \mathcal{C}_i) \geq \lambda_1(\mathcal{C})$ for all $1 \leq i \leq n$ and $j \in \mathbb{F}_q^*$. Because the input is a NO instance, $\lambda_1(\mathcal{C}) > \gamma d$, and so we have that for every $1 \leq i \leq n$, $\lambda_1(\mathcal{C}_i) \geq \lambda_1(\mathcal{C}) > \gamma d$, and therefore $d < \lambda_1(\mathcal{C}_i)/\gamma$. Moreover, for every $1 \leq i \leq n$ and $j \in \mathbb{F}_q^*$, $\mathrm{dist}(\boldsymbol{t}_{i,j}, \mathcal{C}_i) \geq \lambda_1(\mathcal{C}) > \gamma d$. Therefore, each instance $(G_i, \boldsymbol{t}_{i,j}, d)$ is a NO instance of γ-$\mathrm{GapNCP}_q^{1/\gamma}$, as needed. ◀

3.4 NP-hardness of the Code Distortion Problem

We next give a reduction from γ'-GapNCP$_q^\alpha$ to γ-CDP$_q$ for which it suffices to take $\gamma' = O(\gamma)$.

▶ **Theorem 26.** *Let $\gamma \geq 1$, let q be a prime power, and let $D > 1$ be a constant. Then for any $\gamma' > \lceil \frac{D+1}{D-1} \rceil D \cdot \gamma$, there is a Karp reduction from γ'-GapNCP$_q^{1/\gamma'}$ to γ-CDP with distortion D.*

Proof. Let $(G \in \mathbb{F}_q^{n \times k}, \boldsymbol{t} \in \mathbb{F}_q^n, d \in \mathbb{Z}^+)$ be an instance of γ'-GapNCP$_q^{1/\gamma'}$. Define

$$G_1 := \begin{pmatrix} G & 0 & \mathbf{0} \\ 0 & G & \mathbf{0} \\ 0 & 0 & \mathbf{1}_r \end{pmatrix}, \qquad G_2 := \begin{pmatrix} G & 0 & -\boldsymbol{t} \\ 0 & G & \mathbf{0} \\ 0 & 0 & \mathbf{1}_r \end{pmatrix}$$

with $r := \lceil \frac{D+1}{D-1} \rceil \cdot d$. The reduction outputs the γ-CDP$_q$ instance (G_1, G_2, D).

It is clear that the reduction is efficient, and it remains to shows its correctness. Let $\mathcal{C} := \mathcal{C}(G)$, $\mathcal{C}_1 := \mathcal{C}(G_1)$, and $\mathcal{C}_2 := \mathcal{C}(G_2)$. Suppose that the input is a YES instance of γ'-GapNCP$_q^{1/\gamma'}$. Let $\boldsymbol{t}' \in \mathcal{C} - \boldsymbol{t}$ be such that $\|\boldsymbol{t}'\|_0 = \mathrm{dist}(\boldsymbol{t}, \mathcal{C})$, i.e., $\boldsymbol{t}'$ is a shortest codeword in the coset $\mathcal{C} - \boldsymbol{t}$. Define

$$G_2' := \begin{pmatrix} G & 0 & \boldsymbol{t}' \\ 0 & G & \mathbf{0} \\ 0 & 0 & \mathbf{1}_r \end{pmatrix}$$

Notice that $\mathcal{C}(G_2) = \mathcal{C}(G_2')$, and, because the input is a YES instance, that $\|\boldsymbol{t}'\|_0 \leq d$.[10] Let $G_1 = (\boldsymbol{g}_1, \ldots, \boldsymbol{g}_{2n+1})$, let $G_2' = (\boldsymbol{g}_1', \ldots, \boldsymbol{g}_{2n+1}')$, and let M be an invertible linear map such that $M : \boldsymbol{g}_i \mapsto \boldsymbol{g}_i'$ for every i. Such a full-rank map M exists because G_1 and G_2' are full-rank, and it is clear that M maps $\mathcal{C}_1$ to $\mathcal{C}_2$.

Let $\boldsymbol{x} \in \mathcal{C}_1$ be an arbitrary non-zero codeword. We can write $\boldsymbol{x} = (\boldsymbol{x}', \boldsymbol{x}'', a\mathbf{1}_r)$ for some $\boldsymbol{x}', \boldsymbol{x}'' \in \mathcal{C}$ and $a \in \mathbb{F}_q$. If $a = 0$, then $\|M\boldsymbol{x}\|_0 = \|\boldsymbol{x}\|_0$, and so $\|M\boldsymbol{x}\|_0 / \|\boldsymbol{x}\|_0 = 1$. On the other hand, assume that $a \neq 0$. Then $M\boldsymbol{x} = \boldsymbol{x} + (a\boldsymbol{t}', \mathbf{0})$, and so by triangle inequality and the fact that $\|\boldsymbol{t}'\|_0 \leq d$, $\|M\boldsymbol{x}\|_0 \leq \|\boldsymbol{x}\|_0 + d$. Therefore,

$$\frac{\|M\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} \leq \frac{\|\boldsymbol{x}\|_0 + d}{\|\boldsymbol{x}\|_0} \leq \frac{\|\boldsymbol{x}\|_0 + d}{\|a\mathbf{1}_r\|_0} = 1 + d/r \ . \tag{5}$$

Furthermore, $\boldsymbol{x} = M\boldsymbol{x} - (a\boldsymbol{t}', \mathbf{0})$, and so $\|\boldsymbol{x}\|_0 = \|M\boldsymbol{x} - (a\boldsymbol{t}', \mathbf{0})\|_0 \leq \|M\boldsymbol{x}\|_0 + \|\boldsymbol{t}'\|_0 \leq \|M\boldsymbol{x}\|_0 + d$. It follows that $\|M\boldsymbol{x}\|_0 \geq \|\boldsymbol{x}\|_0 - d$, and therefore,

$$\frac{\|M\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} \geq \frac{\|\boldsymbol{x}\|_0 - d}{\|\boldsymbol{x}\|_0} = 1 - \frac{d}{\|\boldsymbol{x}\|_0} \geq 1 - \frac{d}{\|a\mathbf{1}_r\|_0} = 1 - d/r \ . \tag{6}$$

Recalling that $r = \lceil \frac{D+1}{D-1} \rceil d \geq \frac{D+1}{D-1} d$, we get that $d/r \leq \frac{D-1}{D+1}$. By combining Equations (5)

[10] Actually computing $\boldsymbol{t}'$ amounts to solving γ'-GapNCP$_q^{1/\gamma'}$, which is a hard problem. However, here we are only using $\boldsymbol{t}'$ and G_2' for analysis, and we are not computing them.

and (6) we then have that

$$\begin{aligned}\mathcal{D}(\mathcal{C}_1,\mathcal{C}_2) &\leq \max_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}}\left(\frac{\|M\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\right)\Big/\min_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}}\left(\frac{\|M\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\right)\\ &\leq (1+d/r)\,/\,(1-d/r)\\ &\leq \left(1+\frac{D-1}{D+1}\right)/\left(1-\frac{D-1}{D+1}\right)\\ &= \left(\frac{2D}{D+1}\right)/\left(\frac{2}{D+1}\right)\\ &= D,\end{aligned}$$

as needed.

Now, suppose that the input is a NO instance, and let M be an invertible linear map such that $M(\mathcal{C}_1)=\mathcal{C}_2$. Let $\boldsymbol{y}:=M\boldsymbol{g}_{2n+1}=M\cdot(\mathbf{0},\mathbf{1}_r)$, and note that $\boldsymbol{y}=(\boldsymbol{y}'-a\boldsymbol{t},\boldsymbol{y}'',a\mathbf{1}_r)$ for some $\boldsymbol{y}',\boldsymbol{y}''\in\mathcal{C}$ and $a\in\mathbb{F}_q$. If $a\neq 0$, then

$$\|\boldsymbol{y}\|_0=\|\boldsymbol{y}'-a\boldsymbol{t}\|_0+\|\boldsymbol{y}''\|_0+\|a\mathbf{1}_r\|_0\geq \operatorname{dist}(-a\boldsymbol{t},\mathcal{C})+r>\gamma' d+r\ ,$$

where we have used that $\operatorname{dist}(-a\boldsymbol{t},\mathcal{C})=\operatorname{dist}(\boldsymbol{t},\mathcal{C})$ for $a\neq 0$. On the other hand, if $a=0$, then

$$\|\boldsymbol{y}\|_0=\|\boldsymbol{y}'\|_0+\|\boldsymbol{y}''\|_0\geq\lambda_1(\mathcal{C})>\gamma' d\ ,$$

where the first inequality holds because at least one of $\boldsymbol{y}'$ and $\boldsymbol{y}''$ is a non-zero codeword in $\mathcal{C}$ (since $\boldsymbol{y}\neq\mathbf{0}$), and the second inequality holds because of the assumption that $d<\lambda_1(\mathcal{C})/\gamma'$, which follows from the definition of NO instances of γ'-GapNCP$_q^{1/\gamma'}$. In either case, we have that

$$\max_{\boldsymbol{x}\in\mathcal{C}_1\setminus\{\mathbf{0}\}}\frac{\|M\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}\geq\frac{\|M\boldsymbol{g}_{2n+1}\|_0}{\|\boldsymbol{g}_{2n+1}\|_0}=\frac{\|\boldsymbol{y}\|_0}{r}>\gamma' d/r\ . \tag{7}$$

Now, let $\boldsymbol{x}'\in\mathcal{C}$ be such that $\|\boldsymbol{x}'\|_0=\lambda_1(\mathcal{C})$, and define $\boldsymbol{x}_1:=(\boldsymbol{x}',\mathbf{0},\mathbf{0}),\boldsymbol{x}_2:=(\mathbf{0},\boldsymbol{x}',\mathbf{0})$. Note that $\boldsymbol{x}_1,\boldsymbol{x}_2\in\mathcal{C}_2$, and that $\boldsymbol{y}_1:=M^{-1}\boldsymbol{x}_1=(\boldsymbol{y}_1',\boldsymbol{y}_1'',a_1\mathbf{1}_r)$ and $\boldsymbol{y}_2:=M^{-1}\boldsymbol{x}_2=(\boldsymbol{y}_2',\boldsymbol{y}_2'',a_2\mathbf{1}_r)$ for some $\boldsymbol{y}_1',\boldsymbol{y}_1'',\boldsymbol{y}_2',\boldsymbol{y}_2''\in\mathcal{C}$ and $a_1,a_2\in\mathbb{F}_q$. Furthermore, because $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are linearly independent and M is invertible, $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ must also be linearly independent. So, at least one of $\boldsymbol{y}_1',\boldsymbol{y}_1'',\boldsymbol{y}_2',\boldsymbol{y}_2''$ is non-zero, and therefore $\max\{\|M^{-1}\boldsymbol{x}_1\|_0,\|M^{-1}\boldsymbol{x}_2\|_0\}\geq\lambda_1(\mathcal{C})$. It follows that

$$\begin{aligned}\max_{\boldsymbol{x}\in\mathcal{C}_2\setminus\{\mathbf{0}\}}\frac{\|M^{-1}\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0}&\geq\max\left\{\frac{\|M^{-1}\boldsymbol{x}_1\|_0}{\|\boldsymbol{x}_1\|_0},\frac{\|M^{-1}\boldsymbol{x}_2\|_0}{\|\boldsymbol{x}_2\|_0}\right\}\\ &=\max\{\|M^{-1}\boldsymbol{x}_1\|_0,\|M^{-1}\boldsymbol{x}_2\|_0\}/\lambda_1(\mathcal{C})\geq 1\ .\end{aligned} \tag{8}$$

Combining Equations (7) and (8) then gives

$$\mathcal{D}(\mathcal{C}_1,\mathcal{C}_2)>\gamma' d/r=\frac{\gamma'}{\lceil\frac{D+1}{D-1}\rceil}>\frac{\lceil\frac{D+1}{D-1}\rceil\cdot\gamma D}{\lceil\frac{D+1}{D-1}\rceil}=\gamma D\ ,$$

as needed. ◀

From Theorems 25 and 26 and the fact that γ-GapMDP is NP-hard for any constant $\gamma\geq 1$, we conclude that the CDP is NP-hard to approximate to within any constant factor.

▶ **Theorem 1.** *For any constant $\gamma \geq 1$, any constant $D > 1$, and any prime power $q \leq \text{poly}(n)$, γ-GapCDP$_q$ with distortion threshold D is* NP*-hard under deterministic Cook reductions.*

Proof. Combine the NP-hardness result in Theorem 8 with the reductions in Theorems 25 and 26. ◀

We note that restriction on the field size q in Theorem 1 is due the running time of the reduction in Theorem 25.

4 Algorithms for Code Distortion

We now turn to giving algorithms for code distortion.

4.1 An Exact Algorithm

We first analyze the running time of a brute force exact algorithm for CDP, which is the best exact algorithm we know.

▶ **Lemma 27.** *Let $n, k \in \mathbb{Z}^+$, $k \leq n$, let q be a prime power, and let $\mathcal{C}_1, \mathcal{C}_2$ be $[n,k]_q$ codes. The distortion $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$ can be computed in $O^*(q^{k(k+1)})$ time.*

Proof. A linear transformation T with $T\mathcal{C}_1 = \mathcal{C}_2$ must map a generator matrix of $\mathcal{C}_1$ to a generator matrix of $\mathcal{C}_2$. So, it suffices to fix a generator matrix of $G_1 \in \mathbb{F}_q^{n\times k}$, enumerate generator matrices $G_2 \in \mathbb{F}_q^{n\times k}$ of $\mathcal{C}_2$, and compute $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2)$ for the induced map T such that $TG_1 = G_2$.

To enumerate generator matrices G_2 of $\mathcal{C}_2$, it suffices to compute $G_2 := G_2'U$ for a fixed generator matrix G_2' of $\mathcal{C}_2$ and each $U \in \text{GL}_k(\mathbb{F}_q)$. In order to enumerate elements of $\text{GL}_k(\mathbb{F}_q)$, it suffices to enumerate all matrices $U \in \mathbb{F}_q^{k\times k}$, and discard them if they are singular. This takes $O^*(q^{k^2})$ time.

To compute $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2)$ for a given map T, it suffices to compute $\|T\boldsymbol{x}\|_0/\|\boldsymbol{x}\|_0$ for all $\boldsymbol{x} \in \mathcal{C}_1 \setminus \{\mathbf{0}\}$ and $\|T^{-1}\boldsymbol{y}\|_0/\|\boldsymbol{y}\|_0$ for all $\boldsymbol{y} \in \mathcal{C}_2 \setminus \{\mathbf{0}\}$. This can be done in $O^*(q^k)$ time since $|\mathcal{C}_1| = |\mathcal{C}_2| = q^k$. So, overall the algorithm runs in $O^*(q^{k(k+1)})$, as needed. ◀

We then get the following corollary.

▶ **Corollary 28.** *Let $n, k \in \mathbb{Z}^+$, $k \leq n$, let q be a prime power, and let $\mathcal{C}_1, \mathcal{C}_2$ be $[n,k]_q$ codes. The distortion $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$ can be computed in* poly(n) *time if $k = O\left(\sqrt{\log_q n}\right)$.*

Notably, Lemma 27 runs in roughly q^{k^2} time, and at a minimum it would be desirable to find a single-exponential, roughly q^k-time algorithm. Although we do not give an exact such q^k-time algorithm, we do give a roughly q^k-time *approximation* algorithm for CDP$_q$.

4.2 A General Approximation Algorithm

Our approximation algorithm uses the following fact.

▶ **Lemma 29.** *Let $n, k \in \mathbb{Z}^+, k \leq n$, let q be a prime power, and let $\mathcal{C}_1, \mathcal{C}_2$ be $[n,k]_q$ codes with successive minima bases $G_1 = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k), G_2 = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k)$, respectively, and let T be a linear transformation such that $TG_1 = G_2$. Furthermore, let $\boldsymbol{x} = G_1\boldsymbol{a}$ for $\boldsymbol{a} \in \mathbb{F}_q^k$ be a non-zero codeword in $\mathcal{C}_1$. Suppose that $j \in [k]$ is the maximum index such that $a_j \neq 0$. Then*

$$\frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} \leq \frac{\sum_{i=1}^{j} \lambda_i(\mathcal{C}_2)}{\lambda_j(\mathcal{C}_1)} .$$

Proof. Since $TG_1 = G_2$, we know that $T\boldsymbol{v}_i = \boldsymbol{w}_i$ for $1 \leq i \leq k$ and that $T\boldsymbol{x} = \sum_{i=1}^{j} T(a_i\boldsymbol{v}_i) = \sum_{i=1}^{j} a_i\boldsymbol{w}_i$. Using the triangle inequality, we get the upper bound $\|T\boldsymbol{x}\|_0 \leq \sum_{i=1}^{j}\|a_i\boldsymbol{w}_i\|_0 \leq \sum_{i=1}^{j}\|\boldsymbol{w}_i\|_0 = \sum_{i=1}^{j}\lambda_i(\mathcal{C}_2)$.

On the other hand, the j vectors $\boldsymbol{x}$ and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{j-1}$ must be linearly independent because $a_j \neq 0$ and $\boldsymbol{v}_j \notin \operatorname{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{j-1})$. By the definition of successive minima, we then have that $\|\boldsymbol{x}\|_0 \geq \lambda_j(\mathcal{C}_1)$. Combining this lower bound on $\|\boldsymbol{x}\|_0$ with the upper bound on $\|T\boldsymbol{x}\|_0$ above implies the claim. ◀

We define the following quantity relating the successive minima of $[n,k]_q$ codes $\mathcal{C}_1$ and $\mathcal{C}_2$:

$$M(\mathcal{C}_1, \mathcal{C}_2) \coloneqq \max_{i\in[k]} \frac{\lambda_i(\mathcal{C}_2)}{\lambda_i(\mathcal{C}_1)} \ . \tag{9}$$

We will use this quantity to give lower and upper bounds on the distortion $\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$.

▶ **Theorem 30.** *Let $n, k \in \mathbb{Z}^+, k \leq n$, let q be a prime power, and let $\mathcal{C}_1, \mathcal{C}_2$ be $[n,k]_q$ codes. Let $G_1 = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$ and $G_2 = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k)$ be successive minima bases of $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively, and let T be a linear transformation such that $TG_1 = G_2$ Then*

$$M(\mathcal{C}_1, \mathcal{C}_2)M(\mathcal{C}_2, \mathcal{C}_1) \leq \mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \leq \mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) \leq k^2 \cdot M(\mathcal{C}_1, \mathcal{C}_2)M(\mathcal{C}_2, \mathcal{C}_1) \ .$$

Proof. We first prove the upper bound. For any $\boldsymbol{x} \in \mathcal{C}_1 \backslash \{\boldsymbol{0}\}$, we can write $\boldsymbol{x} = G\boldsymbol{a}$ for some $\boldsymbol{a} \in \mathbb{F}_q^k \setminus \{\boldsymbol{0}\}$. Let $j \in [k]$ be the maximum index such that $a_j \neq 0$. By Lemma 29, we then have that

$$\frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} \leq \frac{\sum_{i=1}^{j}\lambda_i(\mathcal{C}_2)}{\lambda_j(\mathcal{C}_1)} \leq \frac{\sum_{i=1}^{j}\lambda_j(\mathcal{C}_2)}{\lambda_j(\mathcal{C}_1)} \leq k \cdot \frac{\lambda_j(\mathcal{C}_2)}{\lambda_j(\mathcal{C}_1)} \leq k \cdot \max_{i\in[k]} \frac{\lambda_i(\mathcal{C}_2)}{\lambda_i(\mathcal{C}_1)} \ .$$

As this is true for all $x \in \mathcal{C}_1 \backslash \{\boldsymbol{0}\}$, it holds that

$$\max_{\boldsymbol{x}\in\mathcal{C}_1\backslash\{\boldsymbol{0}\}} \frac{\|T\boldsymbol{x}\|_0}{\|\boldsymbol{x}\|_0} \leq k \cdot \max_{i\in[k]} \frac{\lambda_i(\mathcal{C}_2)}{\lambda_i(\mathcal{C}_1)} = k \cdot M(\mathcal{C}_1, \mathcal{C}_2) \ .$$

A symmetric argument applied to T^{-1} yields

$$\max_{\boldsymbol{y}\in\mathcal{C}_2\backslash\{\boldsymbol{0}\}} \frac{\|T^{-1}\boldsymbol{y}\|_0}{\|\boldsymbol{y}\|_0} \leq k \cdot M(\mathcal{C}_2, \mathcal{C}_1) \ ,$$

and plugging both into the definition of distortion yields the upper bound

$$\mathcal{D}(\mathcal{C}_1, \mathcal{C}_2) \leq k^2 \cdot M(\mathcal{C}_1, \mathcal{C}_2)M(\mathcal{C}_2, \mathcal{C}_1) \ .$$

We next prove the lower bound using an analogous proof to one in [8]. Let U be an arbitrary full-rank linear map such that $U\mathcal{C}_1 = \mathcal{C}_2$. Then, because $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_i$ are linearly independent for every $i \in [k]$, $U\boldsymbol{v}_1, \ldots, U\boldsymbol{v}_i$ are also linearly independent. It therefore follows that

$$\lambda_i(\mathcal{C}_2) \leq \max_{j\in[i]}\|U\boldsymbol{v}_j\|_0 \leq \|U|_{\mathcal{C}_1}\|_{0\to 0} \cdot \max_{j\in[i]}\|\boldsymbol{v}_j\|_0 = \|U|_{\mathcal{C}_1}\|_{0\to 0}\lambda_i(\mathcal{C}_1) \ .$$

This yields the bound $\frac{\lambda_i(\mathcal{C}_2)}{\lambda_i(\mathcal{C}_1)} \leq \|U|_{\mathcal{C}_1}\|_{0\to 0}$ for all i, which implies $\max_{i\in[k]} \frac{\lambda_i(\mathcal{C}_2)}{\lambda_i(\mathcal{C}_1)} \leq \|U|_{\mathcal{C}_1}\|_{0\to 0}$. A symmetrical argument with $(U^{-1})|_{\mathcal{C}_2}$ mapping successive minima vectors of $\mathcal{C}_2$ to $\mathcal{C}_1$ yields $\max_{i\in[n]} \frac{\lambda_i(\mathcal{C}_1)}{\lambda_i(\mathcal{C}_2)} \leq \|U^{-1}|_{\mathcal{C}_2}\|_{0\to 0}$, and multiplying the bounds together shows that

$$\left(\max_{i\in[n]} \frac{\lambda_i(\mathcal{C}_2)}{\lambda_i(\mathcal{C}_1)}\right)\left(\max_{i\in[n]} \frac{\lambda_i(\mathcal{C}_1)}{\lambda_i(\mathcal{C}_2)}\right) \leq \|U|_{\mathcal{C}_1}\|_{0\to 0}\|U^{-1}|_{\mathcal{C}_2}\|_{0\to 0}.$$

Because U was chosen arbitrarily, we then have that $M(\mathcal{C}_1, \mathcal{C}_2)M(\mathcal{C}_2, \mathcal{C}_1) \leq \mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$. ◀

We now restate and prove Theorem 2, which gives our main approximation algorithm for CDP_q.

▶ **Theorem 2.** *For any prime power q, there is an algorithm for k^2-*CDP *on $[n,k]_q$ codes that runs in $O^*(q^k)$ time and $\mathrm{poly}(n, \log q)$ space.*

Proof. The algorithm works by computing successive minima bases G_1 and G_2 for $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively, using Lemma 6, and then computing and outputting a linear transformation T such that $TG_1 = G_2$. By Lemma 6, computing G_1 and G_2 takes $O^*(q^k)$ time and polynomial space, and using these bases it is efficient to compute T. Furthermore, $\mathcal{D}_T(\mathcal{C}_1, \mathcal{C}_2) \leq k^2 \cdot \mathcal{D}(\mathcal{C}_1, \mathcal{C}_2)$ by Theorem 30. The theorem follows. ◀

References

1. Martin R. Albrecht, Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, Tanja Lange, Varun Maram, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Kenneth G. Paterson, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Cen Jung Tjhai, Martin Tomlinson, , and Wen Wang. Classic McEliece, 2022. NIST Post-Quantum Cryptography Standardization Project submission.
2. Sanjeev Arora, László Babai, Jacques Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *J. Comput. Syst. Sci.*, 54(2):317–331, 1997. Preliminary version in FOCS 1993. doi:10.1006/JCSS.1997.1472.
3. Per Austrin and Subhash Khot. A simple deterministic reduction for the gap minimum distance of code problem. *IEEE Trans. Inf. Theory*, 2014. Preliminary verison in ICALP 2011. doi:10.1109/TIT.2014.2340869.
4. László Babai, Paolo Codenotti, Joshua A. Grochow, and Youming Qiao. Code equivalence and group isomorphism. In *SODA*, 2011.
5. Alessandro Barenghi, Jean-François Biasse, Edoardo Persichetti, and Paolo Santini. LESS-FM: fine-tuning signatures from the code equivalence problem. In *PQCrypto*, 2021.
6. Alessandro Barenghi, Jean-François Biasse, Edoardo Persichetti, and Paolo Santini. On the computational hardness of the code equivalence problem in cryptography. *Adv. Math. Commun.*, 17(1):23–55, 2023. doi:10.3934/AMC.2022064.
7. Huck Bennett, Drisana Bhatia, Jean-François Biasse, Medha Durisheti, Lucas LaBuff, Vincenzo Pallozzi Lavorante, and Philip Waitkevich. Asymptotic improvements to provable algorithms for the code equivalence problem. *IEEE Transactions on Information Theory*, 72(2):1093–1108, 2026. doi:10.1109/TIT.2025.3637683.
8. Huck Bennett, Daniel Dadush, and Noah Stephens-Davidowitz. On the lattice distortion problem. In *ESA*, 2016.
9. Huck Bennett and Kaung Myat Htay Win. Relating code equivalence to other isomorphism problems. *Designs, Codes and Cryptography*, 93(3):701–723, December 2024. doi:10.1007/S10623-024-01542-3.
10. Ward Beullens. Not enough LESS: an improved algorithm for solving code equivalence problems over $\mathbb{F}_q$. In *SAC*, 2020.
11. Vijay Bhattiprolu, Mrinalkanti Ghosh, Venkatesan Guruswami, Euiwoong Lee, and Madhur Tulsiani. Approximability of $p \to q$ matrix norms: Generalized krivine rounding and hypercontractive hardness. In *SODA*, 2019.
12. Jean-François Biasse, Giacomo Micheli, Edoardo Persichetti, and Paolo Santini. LESS is more: Code-based signatures without syndromes. In *AFRICACRYPT*, volume 12174, pages 45–65. Springer, 2020. doi:10.1007/978-3-030-51938-4_3.
13. Qi Cheng and Daqing Wan. A deterministic reduction for the gap minimum distance problem. *IEEE Trans. Inf. Theory*, 58(11):6935–6941, 2012. Preliminary version in STOC 2009. doi:10.1109/TIT.2012.2209198.

14 Thomas Debris-Alazard, Léo Ducas, and Wessel P. J. van Woerden. An algorithmic reduction theory for binary codes: LLL and more. *IEEE Trans. Inf. Theory*, 68(5):3426–3444, 2022. doi:10.1109/TIT.2022.3143620.

15 Léo Ducas and Shane Gibbons. Hull attacks on the lattice isomorphism problem. In *PKC*, 2023.

16 Ilya Dumer, Daniele Micciancio, and Madhu Sudan. Hardness of approximating the minimum distance of a linear code. *IEEE Trans. Inf. Theory*, 49(1):22–37, 2003. Preliminary version in FOCS 1999. doi:10.1109/TIT.2002.806118.

17 Philippe Gaborit, Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Edoardo Persichetti, Gilles Zémor, Jurjen Bos, Arnaud Dion, Jérôme Lacan, Jean-Marc Robert, Pascal Véron, Paulo L. Barreto, Santosh Ghosh, Shay Gueron, Tim Güneysu, Rafael Misoczki, Jan Richter-Brokmann, Nicolas Sendrier, Jean-Pierre Tillich, and Valentin Vasseur. HQC cryptosystem specification. https://pqc-hqc.org/doc/hqc_specifications_2025_08_22.pdf, 2025. Version dated 2025-08-22.

18 Surendra Ghentiyala and Noah Stephens-Davidowitz. More basis reduction for linear codes: Backward reduction, BKZ, slide reduction, and more. In *APPROX*, 2024.

19 Oded Goldreich, Daniele Micciancio, Shmuel Safra, and Jean-Pierre Seifert. Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. *Inf. Process. Lett.*, 71(2):55–61, 1999. doi:10.1016/S0020-0190(99)00083-6.

20 Alexander Golovnev and Noah Stephens-Davidowitz. Personal communication, 2026.

21 J. Leon. Computing automorphism groups of error-correcting codes. *IEEE Transactions on Information Theory*, 28(3):496–511, 1982. doi:10.1109/TIT.1982.1056498.

22 Jiří Matoušek. Lecture notes on metric embeddings. Lecture notes, Charles University, 2013. Available at https://kam.mff.cuni.cz/~matousek/ba-a4.pdf.

23 Robert J. McEliece. A public-key cryptosystem based on algebraic coding theory, 1978. DSN Progress Report.

24 Daniele Micciancio. Locally dense codes. In *CCC*, 2014.

25 Bryant Morrell. Adventures with code distortion: Hardness and approximation. Master's thesis, University of Colorado Boulder, 2026.

26 National Institute of Standards and Technology (NIST). Post-Quantum Cryptography: Additional Digital Signature Schemes — Round 2 Additional Signatures. https://csrc.nist.gov/projects/pqc-dig-sig/round-2-additional-signatures, 2025.

27 Julian Nowakowski. An improved algorithm for code equivalence. In *PQCrypto*, pages 71–103, 2025. doi:10.1007/978-3-031-86599-2_3.

28 E. Petrank and R.M. Roth. Is code equivalence easy to decide? *IEEE Transactions on Information Theory*, 43(5):1602–1604, 1997. doi:10.1109/18.623157.

29 Oded Regev, 2014. Personal communication.

30 Nicolas Sendrier. Finding the permutation between equivalent linear codes: The support splitting algorithm. *IEEE Trans. Inf. Theory*, 46(4):1193–1203, 2000. doi:10.1109/18.850662.

31 Amnon Ta-Shma. Explicit, almost optimal, epsilon-balanced codes. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 238–251, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3055399.3055408.

Online Matching with Size-Based and Convex Delays

Junhao Gan
School of Computing and Information Systems, The University of Melbourne, Australia

Xiao Sun
School of Computing and Information Systems, The University of Melbourne, Australia

Seeun William Umboh
School of Computing and Information Systems, The University of Melbourne, Australia
ARC Training Centre in Optimisation Technologies, Integrated Methodologies, and Applications (OPTIMA), Melbourne, Australia

Abstract

We study the online min-cost perfect matching with delay (MPMD) problem where m requests arrive in a metric space of n points. In MPMD, an algorithm can choose to match a request or to delay, and the objective is to minimise the sum of connection and delay costs. The connection cost of a match is the distance between the locations of two matched requests in the metric, and the increase of the delay cost is a function of the set of unmatched requests at every moment. In this paper, we study two different types of delay functions, size-based (MPMD-Size) and convex delays (MPMD-Convex).

The study of MPMD-Size was initiated by Deryckere and Umboh (APPROX/RANDOM 2023) where the instantaneous delay increment is a non-negative monotone function of the number of unmatched requests. We give an exponential improvement on the lower bounds for the deterministic and randomized competitive ratios. We also give improved upper bounds in terms of n, as opposed to Deryckere and Umboh's upper bounds that were functions of m, which can be much larger than n. Our results settle the deterministic competitive ratio (up to constants). At the heart of these results is a succinct encoding scheme of MPMD-Size on a given n-point metric as a metrical task system problem on a 2^{n-1}-point metric.

We also consider MPMD-Convex proposed by Liu et al. (ISAAC 2018) where the delay cost incurred by each request is a uniform convex delay function of the time difference between its arrival time and the moment that it is matched by the algorithm. They focused on delay functions f that are unbounded, non-decreasing, continuous, and satisfy $f(0) = f'(0) = 0$, and showed that the deterministic competitive ratio is $\Omega(n)$ for n-point uniform metrics. We show that, surprisingly, when f is a non-negative, monotone polynomial with $f'(0) > 0$, there is an $O(1)$-competitive deterministic algorithm for uniform metrics. Our result completes our understanding of MPMD-Convex on uniform metrics for a broad class of functions.

2012 ACM Subject Classification Theory of computation → Online algorithms

Keywords and phrases online algorithms with delays, competitive analysis, matching

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.25

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2607.00536

Funding *Junhao Gan*: Supported in part by the Australian Government through the Australian Research Council ARC DP230102908.
Xiao Sun: Supported by the Australian Government through the Australian Research Council DP240101353 and by the University of Melbourne through the Melbourne Research Scholarship.
Seeun William Umboh: Supported by the Australian Government through the Australian Research Council DP240101353.



© Junhao Gan, Xiao Sun, and Seeun William Umboh;
licensed under Creative Commons License CC-BY 4.0



1 Introduction

In recent years, there has been much interest in revisiting classical online problems through the lens of online problems with delay (e.g. [3, 17, 13]). In such problems, instead of serving every request immediately, one can choose to delay the service of a request while accumulating a delay cost, and the total cost of a solution is the cost of serving all requests plus the accumulated delay costs. In such problems, the trade-off between service cost and delay cost is at the center of the decision making of an online algorithm: the algorithm can wait until many requests have arrived and then apply an offline algorithm to serve them cheaply as a batch, but this will incur a large delay cost.

In this paper, we study the Min-Cost Perfect Matching with Delay (MPMD) problem proposed by Emek et al. [16]. In MPMD, there is an underlying metric space of n points in which m requests arrive over time at different points. At every moment when there is an available request, the online algorithm can choose to either match it with another request and incur a *connection cost* equal to the distance between the locations of two requests, or to leave it unmatched and incur a *delay cost*.

MPMD has many intuitive interpretations and real-life applications. For example, suppose that one needs to design a mechanism for an online game platform which always needs to match two players to compete against each other. The mechanism wants to match two players who have similar levels to each other as this could reduce their dissatisfaction, which is a connection cost that the platform wants to minimize. The platform is allowed to choose not to match a player immediately but to wait for another player whose level is close enough to the first player to join the game and then to match them, so that the connection cost can be reduced. However, if one player has been waiting too long, they might lose patience and become more and more unsatisfied with this game platform, which will incur more delay cost instead. Therefore, the online algorithm needs to balance the trade-off between these two types of costs, which is at the center of studies in this area. This kind of problems are inherently online since we do not know what will come in the future, and once we have made a decision, it is irreversible.

Delay costs. While the connection cost of a match is the distance between the locations of the two requests that are matched, there are various delay models according to how the delay cost increases at every moment when there is at least one request pending. Some delay models apply a uniform delay function that the total delay of each request is a function of the time different between its arrival and being matched, e.g. linear [16], convex [20], and concave delays [5]. The most general framework is that the delay increment is a function of the set of requests that are pending at every moment and the algorithm is non-clairvoyant that it does not know future delay functions, e.g. set and size-based delays [14].

Competitive ratios: n vs m. Prior studies on these delay models have achieved results that express competitive ratios as functions of n or m. Note that m can be incomparably larger than n when the request sequence is sufficiently long, and such algorithms can perform badly even on a simple one-point metric when m tends to infinity (e.g. [4]). Thus, we aim at designing algorithms whose competitive ratios depend only on n.

1.1 Our Contributions

In this paper, we make contributions to the study of two generalizations of linear delay functions mentioned above that are much less understood than the others: size-based and convex delays.

Size-based delay (MPMD-Size). In the size-based delay setting, at every step the delay increment is a monotone function of the number of currently pending requests (See Section 3.1 for a formal definition). We prove bounds on the deterministic and randomized competitive ratios in terms of n. Previously, only upper bounds in terms of m are known.

▶ **Theorem 1.1.** *For every n-point metric, the deterministic competitive ratio of MPMD-Size is $\Theta(2^n)$.*

▶ **Theorem 1.2.** *For every n-point metric, the randomized competitive ratio (against an oblivious adversary) of MPMD-Size is $O(n^2)$ and $\Omega(n)$.*

We emphasize that the lower bounds in Theorems 1.1 and 1.2 are universal lower bounds: they apply to *every* metric space.

Table 1 Comparisons between the results from [14] and our new results for MPMD-Size. The lower bounds of [14] only apply to uniform metrics while ours apply to every metric.

Competitive Ratio	Results from [14]	Our Results
Deterministic	$O(2^m)$ and $\Omega(n)$	$\Theta(2^n)$
Randomized	$O(m^4)$ and $\Omega(\log n)$	$O(n^2)$ and $\Omega(n)$

Theorems 1.1 and 1.2 improve upon previous results on size-based delays by Deryckere and Umboh [14] in two ways. First, the previous upper bounds for both deterministic and randomized algorithms are expressed as functions of m, the number of requests, while our results are expressed as functions of n, the number of points in the given metric space, and thus is an improvement in the more interesting regime where $m \gg n$. Second, we also provide a reduction from MTS to MPMD-Size while theirs is only from MPMD-Size to MTS, which exponentially improves lower bounds for both deterministic and randomized algorithms on *every* n-point metric. In contrast, their lower bound only holds for uniform metrics. Our results settle the deterministic competitive ratio, up to a constant. See Table 1 for a detailed comparison between our results and those by Deryckere and Umboh.

Convex delay (MPMD-Convex). We also consider the convex delays on n-point uniform metrics, first studied by Liu et al. [20]. Convex delay is applicable to a lot of real-life scenarios as well, e.g., a gaming platform where players think it is acceptable to wait for a short period yet a longer delay can be more and more frustrating and even affect the player's willingness to join the platform again. In this problem, there is a non-decreasing delay function f associated with every request and the delay cost of request r incurred until t is $f(t - a(r))$, where $a(r)$ is its arrival time, and the total delay cost is the sum of every request's individual delay. The previous study by Liu et al. focused on the setting where f is non-decreasing, continuous, unbounded and satisfy $f(0) = f'(0) = 0$. They gave a deterministic lower bound of $\Omega(n)$ and a matching upper bound when f is also a monomial.

We consider polynomials f such that $f'(0) > 0$. Prior to our work, the only known result is that when f is linear, the deterministic competitive ratio is $O(1)$ [2]. Our second main result fills this gap and completes our understanding of the landscape of the competitive ratio of MPMD under uniform convex monotone polynomial delays on uniform metrics.

▶ **Theorem 1.3** (Informal; see Theorem 4.1)**.** *For any fixed convex monotone polynomial delay f such that $f'(0) > 0$, there is an $O(1)$-competitive deterministic algorithm for MPMD under delay f on all n-point uniform metrics.*

Surprisingly, for different delay functions, their local derivatives at $t = 0$ can totally change the competitiveness, even when they have the same superlinear asymptotic behavior when $t \to \infty$. For example, the deterministic competitive ratio is $\Omega(n)$ for $f(t) = t^2$, yet a minor perturbation to $f(t) = (t + \varepsilon)^2 - \varepsilon^2$ by any small $\varepsilon > 0$ completely changes it to $O(1)$. We will see in the proof that it is due to the approximability between $f(t)$ and the linear delay $g(t) = t$: as long as $f(0) = f'(0) = 0$, the ratio between f and g tends to 0 when t tends to 0 and so we cannot use g to approximate f within positive constant factors. The $O(1)$-competitive result applies to a broader class of functions beyond polynomials as well, even some exponential functions; see Remark 4.18 for further discussion.

1.2 Our Techniques

MPMD-Size. We employ a similar high-level approach as Deryckere and Umboh [14] of encoding MPMD-Size as a *metrical task system* (MTS) problem (defined in Section 2). Our key contribution is a significantly more efficient encoding.

The main idea is to define the MTS states so that matching requests in the MPMD-Size instance corresponds to a state transition in the MTS instance, and at each timestep, the instantaneous delay increment in the MPMD-Size instance can be realized by a cost vector in the MTS instance. Deryckere and Umboh used the obvious encoding in which there is an MTS state for each even-sized subset of the requests that have arrived so far, and the current MTS state is the set of requests that have been matched so far. While this makes it trivial to encode the instantaneous delay increment as an MTS cost vector – the cost of a state is exactly the instantaneous delay increment of the requests that have arrived but are not matched – the total number of MTS states is the number of all possible even-sized subsets of the requests, which is exponential in m. It also complicates matters when applying MTS algorithms as the MTS state space grows as new requests arrive.

Our key insight that enables a more succinct encoding of MPMD-Size as MTS is the following. It never hurts to match two pending requests if they are co-located and so the number of unmatched requests at each point is either 0 or 1. Thus, to determine the number of unmatched requests (which in turn determines the instantaneous delay increment), it suffices to keep track for each point, whether there is an unmatched request at that point. This, in turn, can be done by keeping track of the parity of the number of times a request on that point has been previously matched (see the discussion following Lemma 3.1). Therefore, we can simply express the current state of an MPMD-Size algorithm using an n-dimensional Boolean vector. Thus, the total number of MTS states is $O(2^n)$.

Finally, our reductions between MTS and MPMD-Size go in both directions. This lets us improve the lower bounds for both deterministic and randomized algorithms exponentially, and in particular, settle the deterministic competitive ratio (up to constants).

MPMD-Convex on uniform metrics. Different from the linear delay, a convex delay function has an unbounded increase rate when it has been pending unmatched for enough long time. As a result, a reasonable algorithm should avoid this case, otherwise even a short period of waiting may lead to huge delay increase. Therefore, we introduce the concept of T-impatient algorithms, where a request that has been pending for time T must be matched immediately once there is another available request anywhere in the metric space. In this way, we naturally divide the time axis into two parts according to whether there is a request that has been pending for longer than time T at the moment. If there is no request that has been pending for longer than time T, then the delay increase rate of every request is upper and lower bounded by two positive constants (to guarantee the positive lower bound, the property that

$f'(0) > 0$ is necessary) and we can use the linear delay to approximate it up to a constant factor. As for the other case, based on the T-impatient property of the algorithm, the only reason that a request has been pending that long is because there is neither another pending request nor the arrival of a new request to be matched with, which means that there must be a request continuously pending throughout this time period under the optimal solution as well, which provides a lower bound of the optimal cost. Luckily, for all monotone polynomials, such a lower bound is within a constant factor of the delay incurred under the algorithm.

Therefore, to design an algorithm for monotone polynomial delays such that $f'(0) > 0$ on uniform metrics, it can be reduced to designing a T-impatient algorithm whose competitive ratio is a constant for linear delay on uniform metrics. We show that the seminal algorithm for MPMD on tree metrics by Azar et al. [2] does not satisfy the T-impatient property while being implemented on uniform metrics, and thus propose a new algorithm and prove it has the desired property.

1.3 Related Works

Linear delay. When MPMD was first introduced by Emek et al. [16], the delay function they studied is the uniform linear delay. They proposed a randomized algorithm with competitive ratio $O(\log^2 n + \log \Delta)$, where n is the number of points in the metric space and Δ is its aspect ratio. Azar et al. [2] improved the upper bound of the randomized competitive ratio for linear delays to $O(\log n)$ by designing an algorithm based on the randomized tree embedding technique into weighted HSTs from [6]. The current best lower bound for the uniform linear delay function is $\Omega(\log n / \log \log n)$ of any randomized algorithm from [1], which shows a gap between known upper and lower bounds even for the most classical delay setting.

The above results that express competitive ratios in terms of n and Δ usually require the algorithm to know the metric space beforehand, and such competitive ratios are independent of the length of the request sequence. However, there are also results where the competitive ratio is expressed in terms of m, the number of requests in the input sequence, while the algorithm does not need to know the metric space prior to its execution: the first such result was proposed by Bienkowski et al. [9] and their deterministic algorithm is $O(m^{2.46})$-competitive. Bienkowski et al. [8] later improved the result to $O(m)$ by an adaptation of the deterministic primal-dual moat growing technique, and independently and concurrently, Azar and Fanani [4] devised a sublinear $O(m^{0.59})$-competitive deterministic algorithm. So far, the best result that expresses the competitive ratio as a function of m is an $O(\log^5 m)$-competitive deterministic algorithm by Dufay and Wattenhofer [15].

Concave delay. Azar et al. [5] considered the MPMD problem with uniform concave delays. They designed an $O(1)$-competitive deterministic algorithm for the single point and an $O(\log n)$-competitive randomized algorithm for general n-point metric spaces. Deryckere and Umboh [14] generalized the primal-dual framework from [8] and proposed an $O(m)$-competitive deterministic algorithm for uniform concave delays.

Convex delay. Liu et al. [20] considered the MPMD problem with uniform convex delays on n-point uniform metric spaces. They studied monomial functions $f(t) = t^\alpha$ such that $\alpha > 1$ (where t is the time difference between arrival and being matched for each request). They provided a deterministic $O(n)$-competitive algorithm and showed a matching deterministic lower bound of $\Omega(n)$ for all uniform non-decreasing unbounded continuous delay functions such that $f(0) = f'(0) = 0$. This result demonstrates a gap between the solutions for the cases with linear and convex delays, as the algorithm proposed by Azar et al. [2] achieves

constant competitive ratio on uniform metrics because uniform metrics can be seen as trees of height 1. They also provided a randomized $\Omega(\log n)$ lower bound for delay functions in the same class. There is no result for convex delays beyond uniform metrics yet.

Set and size-based delays. In all of the above studies, there is one uniform delay function $f(t)$ known to the algorithm in advance that applies to every request from the input, suggesting how its delay cost increases as time goes by, and the total delay cost is the sum of delay costs incurred by each request. Deryckere and Umboh [14] studied another model of delay function that is not accumulated request-wise but step-wise: at every time step t, the instantaneous delay cost is incurred as a function of the set of unmatched requests during this step. This set-delay model studies MPMD problem in the least restrictive way, as there is essentially no restriction on the structure of the delay functions and the setting is non-clairvoyant, i.e. the algorithm does not know future delays. Similar non-clairvoyant delay settings have been studied in other online problems with delays, e.g., TCP Acknowledgment Problem [7] and Subadditive Joint Replenishment Problem [19, 18]. Unfortunately, under the most general set-delay setting, every deterministic algorithm has competitive ratio of at least $\Omega(\Delta)$, where Δ is the aspect ratio of the metric space. Instead, Deryckere and Umboh further restricted to the case of size-based delays, which have natural applications in practical settings with service-level agreements such as cloud computing. They showed that there is an $O(2^m)$-competitive deterministic algorithm and an $O(m^4)$-competitive randomized algorithm by a reduction to the classical MTS problem introduced by Borodin et al. [10] and applying the best upper bounds for both deterministic and randomized algorithms by Borodin et al. and Bubeck et al. [12] respectively. They also provided lower bounds in terms of n, which is $\Omega(n)$ for deterministic algorithms and $\Omega(\log n)$ for randomized algorithms. These upper and lower bounds show significant gaps between m and n in the scenarios where a request sequence of unbounded length is generated on a given finite metric.

Paper organization. We start by introducing the formal definition of MTS in Section 2. In Section 3, we show the reductions between MPMD-Size and MTS on both directions and hence complete the proof of Theorems 1.1 and 1.2. We study MPMD-Convex on uniform metrics and prove Theorem 1.3 in Section 4. We conclude by some discussion on the results as well as open problems.

2 Preliminaries

In this section, we introduce the definition and properties of the Metrical Task System problem, which will be used in Section 3 to prove the main results of our paper on MPMD-Size. The formal definitions and notations of MPMD-Size and MPMD-Convex, two different delay models of MPMD that we study, are in Sections 3 and 4 respectively.

The metrical task system (MTS) problem, introduced by Borodin et al. [10], can be defined as a sequence of requests $\{\rho_t\}_{t\geq 1}$ on a given finite metric space (V, d) and an initial state $s_0 \in V$ of the server. Each request $\rho_t : V \to \mathbb{R}_{\geq 0}$ is a non-negative cost function on the state space V, illustrating the cost of the server to process request ρ_t in different points from V. Upon the arrival of request ρ_t at every step $t \geq 1$, the server can be moved to some state $s_t \in V$ to process ρ_t, and this incurs *service cost* of $\rho_t(s_t)$. Since MTS is an online problem, the state s_t at every step t must be output without knowing future requests. The total cost of a solution $\{s_t\}_{t\geq 1}$ that expresses the state s_t where the server processes request

ρ_t at every step $t \geq 1$ consists of service cost and *transition cost*, where the transition cost at every step t is the distance $d(s_{t-1}, s_t)$ of moving to s_t from its previous state s_{t-1}. As a result, the cost of a solution $\{s_t\}_{t\geq 1}$ is defined as:

$$\sum_{t\geq 1}[\rho_t(s_t) + d(s_{t-1}, s_t)].$$

For MTS problems, we say that an online algorithm $\mathcal{A}$ is *lazy* if: at every step t when it moves from s_{t-1} to a different state s_t to process ρ_t, it must hold that $\rho_t(s_{t-1}) > 0$. Intuitively speaking, a lazy algorithm $\mathcal{A}$ only makes a move when its current state incurs positive cost for the upcoming request while always staying still if the service cost of the new request in the current state is zero.

▶ **Lemma 2.1.** *Given any online algorithm $\mathcal{A}$ for MTS, there is an online mechanism that transforms $\mathcal{A}$ to a lazy algorithm without increasing the cost.*

From now on, we can assume that every MTS algorithm that we use is lazy.

Let MTS-Single denote the subclass of MTS problem where every request ρ_t incurs positive cost at exactly one point $v_t \in V$, i.e. $\rho_t(v_t) > 0$ and $\rho_t(v) = 0$ for every $v \in V \setminus \{v_t\}$. When a metric space is given, we use c_{m} to denote the minimum possible worst-case competitive ratio that can be achieved by an algorithm for MTS problem on it, and c_{ms} to denote that can be achieved for MTS-Single problem. The following result is folklore for both deterministic and randomized algorithms:

▶ **Lemma 2.2.** $c_{\mathrm{m}} \leq c_{\mathrm{ms}}$.

The proofs of Lemmas 2.1 and 2.2 can be found in the full version. They will be used in Section 3 to establish the reductions between MPMD-Size and MTS.

3 Reduction between MPMD-Size and MTS

In this section, we first formally define the MPMD problem with size-based delays (MPMD-Size), and then describe the reduction between it and MTS.

3.1 Definition and Property of MPMD-Size

The MPMD problem is defined on a metric space (V, d), where V is the set of points and $d : V \times V \to \mathbb{R}_{\geq 0}$ is the distance function. An online input consists of a sequence of requests that arrive over time. Every request r is defined by two characteristics: its arrival time $a(r)$ and location $\ell(r) \in V$. We view the problem in the discrete way that any request can only arrive at a time step $t \in \mathbb{N}_{>0}$, yet it is possible that multiple requests arrive at the same time step t. This model captures the original definition by Emek et al. [16] when we divide the continuous time axis into discrete steps where a step covers an interval of infinitesimal length.

Suppose Q_t is the set of requests that have arrived up to and including time step t. At every step, the instantaneous delay function $f_t : 2^{Q_t} \to \mathbb{R}_{\geq 0}$ specifies the increment of delay cost at this step for every possible $P_t \subseteq Q_t$, where P_t is the set of requests that are pending during time step t. At every step t, the algorithm first sees the newly-arriving requests (if any) and the instantaneous delay function f_t, then makes a decision on matching some requests. A match between two requests r_1 and r_2 incurs a connection cost of $d(\ell(r_1), \ell(r_2))$. This process results in an updated P_t, and the instantaneous delay $f_t(P_t)$ is then incurred

after the matching performed at step t. Observe that the definition is consistent with metrical task systems (defined in Section 2) in the sense that at every step, the algorithm first sees the new input, then decides which state to move to (possibly the same state), and incurs the cost of the resulting state. The goal is to minimize the sum of its connection cost and delay cost incurred among all steps.

In the MPMD-Size problem, the instantaneous delay function incurred at time step t can be expressed as $f_t(|P_t|)$, where $f_t : \mathbb{N}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a monotone function. We note that the well-studied linear delay cost is a special case of the size-based delay where $f_t(|P_t|) = |P_t|$ at every moment t.

We say that an online algorithm $\mathcal{A}$ is *smart* if no two requests are pending at the same point during any time step. One important property of sized-based delay functions is that we can without loss of generality assume that every algorithm or solution is smart, because of the following lemma whose proof can be found in the full version:

▶ **Lemma 3.1.** *For size-based delays, given any online algorithm $\mathcal{A}$, there is an online mechanism that transforms $\mathcal{A}$ to a smart algorithm without increasing the cost.*

Lemma 3.1 provides us with not only a nice property of the optimal solution, but also a mechanism of online transformations that allows us to always automatically match two requests at the same point immediately at any moment. As a result, we do not need to explicitly describe the a match of two requests at the same point in the metric anymore. Based on this, we can assume that at every point $v \in V$ there is either one or zero pending request at every time step t, and there are at most n requests pending in the whole metric space V. Moreover, at every given time step, we can refer to a match between two requests at two different points as a match between these two points without ambiguity. Throughout the paper, we may use expressions such as "the algorithm matches point v_1 with point v_2" and "point v has been matched for an odd number of times", where the former refers to a match between requests at these two points v_1 and v_2, and the latter refers to the total number of times where a request at v has been matched with some request at another point. As we have discussed in Section 1.2, this allows us to encode the state of an algorithm with a Boolean vector that has an entry over every point in V, which builds the foundation of our reductions.

3.2 Construction of MTS Metric Space

Before we formally describe the reduction between two problems, we first specify the corresponding metric space $(\mathcal{K}^V, D)$ on which the request sequence for both MTS and MTS-Single problems is based, given a metric space (V, d) for an MPMD-Size instance.

▶ **Definition 3.2.** *Given any finite set V, define $\mathcal{K}^V := \{\mathbf{u} \in \{0,1\}^V \mid \sum_{v \in V} \mathbf{u}(v) = 0 \mod 2\}$, where $\mathbf{u}(v)$ is the entry of $\mathbf{u}$ on a given $v \in V$ for every $\mathbf{u} \in \{0,1\}^V$.*

Observe that $|\mathcal{K}^V| = 2^{n-1}$.

Intuition for $(\mathcal{K}^V, D)$. To have an intuitive understanding of the structure of $(\mathcal{K}^V, D)$ defined on a metric (V, d) and its addition operation, we can give a Boolean variable to every point in V, and see every match between two points in V as an operation that flips the Boolean variable associated with these two points simultaneously. If the default setting is that the Boolean variable associated with every point in V is 0, then $\mathcal{K}^V$ is exactly the set of states that can be attained by applying the matching operations, where each state is a

Boolean vector that specifies the value of the Boolean variable associated with every point in V. With this intuition, if the cost of a match is exactly the distance of the two points involved in it, then the distance between two states in $\mathcal{K}^V$ should be the minimum cost to transform one state into another by applying matching operations.

As a result, we define the concept of the induced matching of every element in $\mathcal{K}^V$ (Definition 3.5) and use it to express the distance D on $\mathcal{K}^V$. We will see that this definition of distance indeed satisfies the min-cost transformation property between every two states (Lemma 3.18).

▶ **Definition 3.3** (Addition operation on $\{0,1\}^V$). *Let $\oplus$ denote the XOR Boolean operation over $\{0,1\}$, i.e. $a \oplus b = 0$ if $a = b$ and $a \oplus b = 1$ if $a \neq b$, for $a, b \in \{0,1\}$. For every $\mathbf{u}_1, \mathbf{u}_2 \in \{0,1\}^V$, $\mathbf{u}_1 \oplus \mathbf{u}_2 := (\mathbf{u}_1(v) \oplus \mathbf{u}_2(v))_{v \in V}$.*

We use $\mathbf{0}$ to denote $(0)_{v \in V}$ and $\mathbf{1}$ to denote $(1)_{v \in V}$. We first note that $\mathcal{K}^V$ is a closed subgroup of $\{0,1\}^V$ under this addition operation.

▷ Claim 3.4. For every $\mathbf{u}_1, \mathbf{u}_2 \in \mathcal{K}^V$, we have $\mathbf{u}_1 \oplus \mathbf{u}_2 \in \mathcal{K}^V$.

Proof. If $\sum_{v \in V} \mathbf{u}_1(v) = \sum_{v \in V} \mathbf{u}_2(v) = 0 \mod 2$, then $\sum_{v \in V} (\mathbf{u}_1 \oplus \mathbf{u}_2)(v) = \sum_{v \in V} (\mathbf{u}_1(v) \oplus \mathbf{u}_2(v)) = \sum_{v \in V} \mathbf{u}_1(v) + \sum_{v \in V} \mathbf{u}_2(v) = 0 \mod 2$. ◁

▶ **Definition 3.5** (Induced matching of an element in $\mathcal{K}^V$). *Given $\mathbf{u} \in \mathcal{K}^V$, let $T_{\mathbf{u}} = \{v \in V \mid \mathbf{u}(v) = 1\}$. Define the* induced matching *of $\mathbf{u}$ to be an arbitrary min-cost perfect matching of points in $T_{\mathbf{u}}$ in the metric space (V, d).*[1] *For every min-cost perfect matching M of $T_{\mathbf{u}}$, we associate the vector $\mathbf{u}$ with M.*

▶ **Definition 3.6** (Distance D on $\mathcal{K}^V$). *Given $\mathbf{u}_1, \mathbf{u}_2 \in \mathcal{K}^V$, define their distance $D(\mathbf{u}_1, \mathbf{u}_2)$ to be the weight of a min-cost perfect matching induced by $\mathbf{u}_1 \oplus \mathbf{u}_2$.*

See the full version for the proof of the following claim.

▷ Claim 3.7. $(\mathcal{K}^V, D)$ is a metric space.

3.3 From MPMD-Size to MTS

Given any instance of MPMD-Size on the metric space (V, d), we first describe the reduction from it to an instance of MTS on the corresponding $(\mathcal{K}^V, D)$, as well as how to translate an online algorithm for MTS to one for the original MPMD-Size.

We introduce some notations to express the state of an MPMD-Size instance under the execution of a given algorithm at step t, including: the total arrivals of requests at different points, and the effect of all matches made by the algorithm so far.

▶ **Definition 3.8.** *Given an algorithm for MPMD-Size on the metric space (V, d), for every step $t \geq 1$, let $R_t \in \{0,1\}^V$ be the Boolean vector such that $R_t(v) = 1$ if and only if the parity of the number of requests that have arrived at $v \in V$ until (including) step t is odd, $M_t \in \mathcal{K}^V$ be the Boolean vector such that $M_t(v) = 1$ if and only if v has been matched for an odd number of times until (including) step t. The initial setting ($t = 0$) is that $R_0 = M_0 = \mathbf{0}$.*

In MPMD-Size, at every step t, new requests arrive and the instantaneous delay function f_t is revealed at the same time, and an online algorithm needs to make decisions to match according to this information. At each step t, we need to handle two situations: when new

[1] A perfect matching exists since $|T_{\mathbf{u}}|$ is even, by the definition of $\mathcal{K}^V$.

requests have arrived at t and before the algorithm has made new decisions yet, and after the algorithm has made decisions at time t. The former determines the matchings that the algorithm can make, and the latter determines the instantaneous cost incurred at the end of step t. With notations introduced in Definition 3.8, it is easy to see that the former state can be expressed as $R_t \oplus M_{t-1}$ and the latter state can be expressed as $R_t \oplus M_t$. With the smart algorithm property and the mechanism shown in Lemma 3.1, it is not hard to see that we can assume that requests at the same point cancel out each other automatically for state $R_t \oplus M_{t-1}$ as well. We summarize the above discussion in the following observation that expresses the number of request at every point with a Boolean vector:

▶ **Observation 3.9.** *Upon the arrivals of new requests at time step t (before new matches are made by the algorithm at step t), the number of pending requests at every point v is equal to $M_{t-1}(v) \oplus R_t(v)$; once the algorithm has made decisions at step t, the number of pending requests at v is equal to $M_t(v) \oplus R_t(v)$ when the delay cost is incurred at step t.*

If $(M_{t-1} \oplus R_t)(v) = 0$, then there is no request pending at v after new arrivals at t, and of course there cannot be any request at v after new matches are performed at t as well, thus leading to the following observation:

▶ **Observation 3.10.** $(R_t \oplus M_{t-1})(v) \leq (R_t \oplus M_t)(v)$ *for every $v \in V$ and time step t.*

For every $\mathbf{u} \in \{0,1\}^V$, we use $|\mathbf{u}|$ to denote the number of 1's among all entries of $\mathbf{u}$, i.e. $|\mathbf{u}| = \sum_{v \in V} \mathbf{u}(v)$. During every step t when the delay cost is incurred, $R_t \oplus M_t$ is the Boolean vector that denotes the state of the input under the algorithm as per Observation 3.9. Based on these notations and the property of size-based delays, the delay cost incurred during step t can be expressed as $f_t(|R_t \oplus M_t|)$.

Construction of MTS instance. Now we are ready to define the MTS instance that we reduce MPMD-Size to. We set the server's initial position of the MTS instance to be $C_0 = \mathbf{0}$. Upon every time step $t \geq 1$, we assign a request vector $(\rho_t(\mathbf{u}))_{\mathbf{u} \in \mathcal{K}^V}$ to the MTS instance according to R_t, the Boolean vector that we have defined to denote the parity of the number of requests that have arrived at each point in V until (and including) t, and f_t, the instantaneous delay cost function of the MPMD-Size instance incurred at step t. More specifically, the cost to process ρ_t at $\mathbf{u}$ is $\rho_t(\mathbf{u}) = f_t(|R_t \oplus \mathbf{u}|)$ for every $\mathbf{u} \in \mathcal{K}^V$.

Let $\mathrm{OPT_m}$ denote the value of the optimal solution of the MTS instance and $\mathrm{OPT_{size}}$ denote that of the MPMD-Size instance.

▶ **Lemma 3.11.** $\mathrm{OPT_{size}} \geq \mathrm{OPT_m}$.

Proof. Given any solution $\mathrm{SOL_{size}}$ of MPMD-Size, we convert it to a solution $\mathrm{SOL_m}$ of MTS without increasing the cost. Upon every step t, let M_t be the sum of Boolean vectors of matches made by $\mathrm{SOL_{size}}$ so far, then M_t is an element of $\mathcal{K}^V$, and $\mathrm{SOL_m}$ moves from state M_{t-1} to M_t (this is feasible at step 1 since at the beginning $M_0 = \mathbf{0}$ and the initial configuration of the MTS instance is given as $C_0 = \mathbf{0}$ as well).

We now compare the cost of two solutions. At every step t, the number of pending requests in the MPMD-Size solution is $|R_t \oplus M_t|$. As a result, the delay cost incurred during step t is $f_t(|R_t \oplus M_t|)$, which is equal to $\rho_t(M_t)$, the cost to process the task ρ_t at M_t, based on our setting of $(\rho_t(\mathbf{u}))_{\mathbf{u} \in \mathcal{K}^V}$. By Lemma 3.18, the connection cost of $\mathrm{SOL_{size}}$ at step t is at least the distance $D(M_{t-1}, M_t)$, which equals the cost of MTS to move from M_{t-1} to M_t. ◀

Translating MTS algorithm to MPMD-Size. We now describe how to transform an algorithm $\mathcal{A}_{\text{m}}$ for the MTS instance to one $\mathcal{A}_{\text{size}}$ for the MPMD-Size instance in an online way. As we see from the proof of Lemma 3.11, there is naturally a one-to-one correspondence between the current MTS state and the current total matching Boolean vector M_t of MPMD-Size, which can be seen as the state of an MPMD-Size algorithm. Therefore, given an online algorithm $\mathcal{A}_{\text{m}}$ for MTS problem on $(\mathcal{K}^V, D)$ that processes request ρ_t at state C_t, intuitively, we want to transit MPMD-Size algorithm to state $M_t = C_t$ as well. However, such a transition is not always feasible, as there is no guarantee that there is an available request at every point that the algorithm needs to match to transit to C_t. If this happens, then we want to update M_{t-1} to M_t that is the closest possible to C_t among all feasible matchings that can be made: we check every pair of match in the min-cost perfect matching induced by $M_{t-1} \oplus C_t$, and match every feasible pair of requests. See the description of Algorithm 1 for more details.

Algorithm 1 The translation algorithm of an online algorithm $\mathcal{A}_{\text{m}}$ for MTS on $(\mathcal{K}^V, D)$ to $\mathcal{A}_{\text{size}}$ for MPMD-Size on (V, d).

Initialize: $M_0 = \mathbf{0}$ for MPMD-Size, and the initial configuration of MTS is $C_0 = \mathbf{0}$.

At every step t:

Given R_t, the arrivals of requests so far in the MPMD-Size instance and f_t the instantaneous size-based delay cost, task vector ρ_t of MTS is constructed such that $\rho_t(\mathbf{u}) = f_t(|R_t \oplus \mathbf{u}|)$ for every $\mathbf{u} \in \mathcal{K}^V$, and we are given $\mathcal{A}_{\text{m}}$'s state C_t at step t.

Upon arrivals of new requests at t, the Boolean vector of current available requests in V is $R_t \oplus M_{t-1}$ (Observation 3.9).

Calculate $M_{t-1} \oplus C_t$, the Boolean vector of the matching that $\mathcal{A}_{\text{size}}$ intends to make to transit to state M_t. For every edge (u, v) in the min-cost perfect matching induced by $M_{t-1} \oplus C_t$, if $(M_{t-1} \oplus R_t)(u) = (M_{t-1} \oplus R_t)(v) = 1$, then there is a pending request at both u and v after new arrivals at step t, and we match u and v in this case.

After all matches have been performed, the new vector M_t is updated accordingly.

▷ Claim 3.12. For every $v \in V$, if $M_t(v) \neq C_t(v)$, then $(M_{t-1} \oplus C_t)(v) = 1$.

Proof. We prove the contrapositive. If $(M_{t-1} \oplus C_t)(v) = 0$, then $\mathcal{A}_{\text{m}}$ does not make any match that involves v at step t, and so $M_t(v) = M_{t-1}(v) = C_t(v) = 0$. ◁

The following claim guarantees the local optimality of $\mathcal{A}_{\text{m}}$ as per the transition from M_{t-1} to M_t by matches at step t:

▷ Claim 3.13. The connection cost of $\mathcal{A}_{\text{m}}$ at step t is equal to $D(M_{t-1}, M_t)$.

Proof. Let J_1 be the set of edges matched by $\mathcal{A}_{\text{m}}$ at step t and J_2 be the set of edges in the min-cost perfect matching induced by $M_{t-1} \oplus C_t$, then $J_1 \subseteq J_2$ by the mechanism of $\mathcal{A}_{\text{m}}$. If J_1' is a another perfect matching on $T_{M_{t-1} \oplus M_t} = \{v \in V \mid (M_{t-1} \oplus M_t)(v) = 1\}$ such that the cost of J_1' is smaller than J_1, then $(J_2 \setminus J_1) \cup J_1'$ is a perfect matching on $T_{M_{t-1} \oplus C_t} = \{v \in V \mid (M_{t-1} \oplus C_t)(v) = 1\}$ whose cost is smaller than J_2, a contradiction to the assumption that J_2 is a min-cost perfect matching! ◁

We prove that the cost of $\mathcal{A}_{\text{size}}$ is no more than that of $\mathcal{A}_{\text{m}}$ by the following lemmas. First, we compare the instantaneous delay cost of $\mathcal{A}_{\text{size}}$ incurred at step t with the cost of processing task ρ_t by $\mathcal{A}_{\text{m}}$, and this is where the size-based property of the delay function f_t plays a vital role.

▶ **Lemma 3.14.** *The delay cost at step t by $\mathcal{A}_{\text{size}}$ is no more than the service cost to process request ρ_t by $\mathcal{A}_{\text{m}}$, given that f_t is a size-based delay.*

Proof. The delay cost of $\mathcal{A}_{\text{size}}$ at step t is $f_t(|R_t \oplus M_t|)$ and the service cost to process request ρ_t by $\mathcal{A}_{\text{m}}$ is $f_t(|R_t \oplus C_t|)$. By the monotonicity of the size-based delay function, it suffices to show that $|R_t \oplus M_t| \leq |R_t \oplus C_t|$. We do this by constructing an injective map $h_t : P'_t \setminus P_t \to P_t \setminus P'_t$, where

$$P'_t = \{v \in V \mid (R_t \oplus M_t)(v) = 1\}; \qquad P_t = \{v \in V \mid (R_t \oplus C_t)(v) = 1\}.$$

For every $v \in P'_t \setminus P_t$, it holds that $M_t(v) \neq C_t(v)$ according to the definition of P'_t and P_t. By Claim 3.12, it must be that $(M_{t-1} \oplus C_t)(v) = 1$. Let u be the request that is matched with u in the min-cost perfect matching induced by $M_{t-1} \oplus C_t$. We now prove that $u \in P_t \setminus P'_t$, and hence we can define $h_t(v) = u$.

By Observation 3.10, $(R_t \oplus M_{t-1})(v) = 1$ for every $v \in P'_t$. By the construction of $\mathcal{A}_{\text{size}}$, if $(R_t \oplus M_{t-1})(v) = 1$ and $M_t(v) \neq C_t(v)$ both hold, then it must be that $(R_t \oplus M_{t-1})(u) = 0$ making v unmatched at step t. In this case, it holds that $(R_t \oplus M_t)(u) \leq (R_t \oplus M_{t-1})(u) = 0$ according to Observation 3.10, which implies that $u \notin P'_t$ by definition. On the other hand, $(M_{t-1} \oplus C_t)(u) = 1$ yet u is unmatched at step t, which implies that $(R_t \oplus C_t)(u) = (R_t \oplus M_{t-1})(u) \oplus 1 = 1$, and so $u \in P_t$. As a result, $u \in P_t \setminus P'_t$ and h_t is well-defined on $P'_t \setminus P_t$.

We now prove that h_t is injective. For any $v_1 \neq v_2$ from $P'_t \setminus P_t$, $h_t(v_1)$ is matched with v_1 and $h_t(v_2)$ is matched with v_2 in the min-cost perfect matching induced by $M_{t-1} \oplus C_t$. As a result, $h_t(v_1) \neq h_t(v_2)$ by the matching property, and h is an injection. ◀

The next lemma compares the connection cost of $\mathcal{A}_{\text{size}}$ (which is $D(M_{t-1}, M_t)$ at every step t according to Claim 3.13) with the transition cost of $\mathcal{A}_{\text{m}}$.

▶ **Lemma 3.15.** *$\sum_{t=1}^{T} D(M_{t-1}, M_t) \leq \sum_{t=1}^{T} D(C_{t-1}, C_t)$, where T is the final step of both of the corresponding MTS and MPMD-Size instances.*

Proof. We define a potential function $\phi_t = D(M_t, C_t)$ for each step t. It is easy to see that $\phi_0 = 0$ and $\phi_t \geq 0$ for every t. We prove the telescopic inequalities

$$D(M_{t-1}, M_t) \leq D(C_{t-1}, C_t) - (\phi_t - \phi_{t-1})$$

for every step $t \geq 1$ to draw the conclusion.

First, by the triangle inequality on the metric space $(\mathcal{K}^V, D)$, it holds that

$$D(M_{t-1}, C_t) \leq D(M_{t-1}, C_{t-1}) + D(C_{t-1}, C_t) = D(C_{t-1}, C_t) + \phi_{t-1}.$$

We now show that $D(M_{t-1}, M_t) + \phi_t = D(M_{t-1}, M_t) + D(C_t, M_t) \leq D(M_{t-1}, C_t)$, then we get the desired inequality by summing up these two.

Define

$$T_1 = \{v \in V \mid (M_{t-1} \oplus M_t)(v) = 1\}; \qquad T_2 = \{v \in V \mid (M_t \oplus C_t)(v) = 1\}.$$

We first observe that $T_{M_{t-1} \oplus C_t} = \{v \in V \mid (M_{t-1} \oplus C_t)(v) = 1\} = T_1 \Delta T_2$, according to the definition of $\oplus$ operation of Boolean vectors. By Claim 3.12, for every $v \in T_2$, it must hold that $v \in T_{M_{t-1} \oplus C_t}$ and hence $v \notin T_1$, which implies that $T_1 \cap T_2 = \varnothing$. As a result, we can express $T_{M_{t-1} \oplus C_t} = T_1 \sqcup T_2$[2] . Moreover, for each pair (u, v) in the min-cost perfect

[2] $T_1 \sqcup T_2$ is the disjoint union of T_1 and T_2.

matching induced by $M_{t-1} \oplus C_t$, either u, v are both in T_1, or u, v are both in T_2, according to the mechanism of $\mathcal{A}_{\text{size}}$ to update M_t at every step. Therefore, we can divide all matches in the min-cost perfect matching induced by $M_{t-1} \oplus C_t$ into two parts: J_1 that forms a matching on T_1, and J_2 that forms a matching on T_2. Using $w(\cdot)$ to denote the weight of a matching, we have $w(J_1) + w(J_2) = D(M_{t-1}, C_t)$ since $J_1 \cup J_2$ forms the min-cost perfect matching induced by $M_{t-1} \oplus C_t$. Moreover, $D(M_{t-1}, M_t) \leq \omega(J_1)$ and $D(M_t, C_t) \leq \omega(J_2)$ by the definition of distance D. This proves the desired inequality. ◀

Combining Lemmas 3.14 and 3.15, we conclude that the total cost of $\mathcal{A}_{\text{size}}$ on the MPMD-Size instance on (V, d) is no more than the total cost of $\mathcal{A}_{\text{m}}$ on the MTS instance on $(\mathcal{K}^V, D)$, yielding the following theorem of the reduction.

▶ **Theorem 3.16.** *Let (V, d) be a metric, c_{m} be the competitive ratio of MTS on $(\mathcal{K}^V, D)$ and c_{size} be that of MPMD-Size on (V, d). Then, $c_{\text{size}} \leq c_{\text{m}}$.*

Applying the upper bound of $O(N)$ for deterministic algorithms from [10] and that of $O(\log^2 N)$ for randomized algorithms from [12] of MTS on every N-point metric gives the upper bounds in Theorems 1.1 and 1.2.

▶ **Corollary 3.17.** *There is an $O(2^n)$-competitive deterministic algorithm and an $O(n^2)$-competitive randomized algorithm for MPMD-Size on every n-point metric.*

3.4 From MTS-Single to MPMD-Size

In this part, we establish the reduction from any given instance of MTS-Single on $(\mathcal{K}^V, D)$ to one of MPMD-Size on (V, d), where V is a given metric space and $|V| = n$.

MPMD-Size delay function. We first state the structure of the size-based delay function f_t of the MPMD-Size instance for every step $t \geq 1$:

$$f_t(N) = \begin{cases} 0 & \text{if } N < n; \\ \varepsilon_t & \text{if } N \geq n. \end{cases}$$

Here ε_t denotes a positive amount in our construction that will depend on the request of the MTS-Single instance at step t. More specifically, at every step t there is exactly one point that is "hit" with a positive cost by the request in MTS-Single, and we use ε_t to denote the amount of this positive cost.

Requests of MPMD-Size instance. We assume that the initial setting of the MTS server is $C_0 = \mathbf{0}$, otherwise we set $R_t \oplus \mathbf{u}_t = C_0 \oplus \mathbf{1}$ and show that the property maintained in the proof of Lemma 3.19 is $M_t \oplus C_t = C_0$ for every step t instead (it is intuitively like "shifting" both R_t and $M_t \oplus C_t$ by C_0). At every step t when the new request ρ_t arrives in the MTS-Single instance on $(\mathcal{K}^V, D)$, suppose that $\rho_t(\mathbf{u}_t) = \varepsilon_t > 0$ and $\rho_t(\mathbf{u}) = 0$ for every $\mathbf{u} \in \mathcal{K}^V \setminus \{\mathbf{u}_t\}$, then we set $R_t = \mathbf{u}_t \oplus \mathbf{1}$ for the MPMD-Size instance. To have a more intuitive understanding of how new requests arrive in the metric space at step t, we use the difference between R_{t-1} and R_t, i.e. $R_{t-1} \oplus R_t$, to denote the new arrivals at step t, such that $(R_{t-1} \oplus R_t)(v) = 1$ if and only if a new request arrives at step t at point v, for every $v \in V$. Moreover, we can express this amount as $R_{t-1} \oplus R_t = R_{t-1} \oplus \mathbf{u}_t \oplus \mathbf{1}$, which can be computed from R_{t-1} that has been well-defined from previous steps, and $\mathbf{u}_t$ defined by the new request of MTS-Single instance at step t. As a result, the MPMD-Size instance that we reduce MTS-Single to is well-defined.

The following lemma provides a lower bound of a solution's connection cost at step t, given M_{t-1} and M_t, the two consecutive states of matches that have been made so far.

▶ **Lemma 3.18.** *The cost of matches made at step t is at least $D(M_{t-1}, M_t)$.*

Proof. Let $M(v)$ denote the number of times that the algorithm matches v with another point at step t, then $M(v)$ is either 0 or 1 based on the assumption that there is at most one available request at every $v \in V$ and so we naturally regard M as a Boolean vector. For every $v \in V$, it holds that $M_t(v) = M_{t-1}(v) \oplus M(v)$, according to the definition of M_t and M_{t-1} and the addition operation on Boolean vectors. As a result, M is a perfect matching of $T_{M_{t-1} \oplus M_t} = \{v \in V \mid (M_{t-1} \oplus M_t)(v) = 1\}$ (since we only consider matches between different points). The statement follows by Definition 3.6. ◀

Let $\mathrm{OPT}_{\mathrm{ms}}$ denote the value of the optimal solution of the MTS-Single instance and $\mathrm{OPT}_{\mathrm{size}}$ denote the that of the MPMD-Size instance.

▶ **Lemma 3.19.** $\mathrm{OPT}_{\mathrm{ms}} \geq \mathrm{OPT}_{\mathrm{size}}$.

See the full version of our paper for the proof of Lemma 3.19.

The next step is to convert an online algorithm $\mathcal{A}_{\mathrm{size}}$ for the MPMD-Size instance to one $\mathcal{A}_{\mathrm{ms}}$ for the MTS-Single instance without increasing the cost. At every step t after new matches have been made by $\mathcal{A}_{\mathrm{size}}$ and M_t is updated, $\mathcal{A}_{\mathrm{ms}}$ processes request ρ_t in state M_t.

▶ **Lemma 3.20.** *The cost of $\mathcal{A}_{\mathrm{ms}}$ is no more than that of $\mathcal{A}_{\mathrm{size}}$.*

Proof. It is easy to see the transition cost of $\mathcal{A}_{\mathrm{ms}}$ is no more than the connection cost of $\mathcal{A}_{\mathrm{size}}$. This is because the matching made by $\mathcal{A}_{\mathrm{size}}$ at t must be at least the value of a min-cost perfect matching induced by $M_t \oplus M_{t-1}$ according to Lemma 3.18, which is equal to $D(M_{t-1}, M_t)$, the cost of moving from the previous state M_{t-1} to a new state M_t at step t in the MTS-Single instance.

On the other hand, the delay cost incurred for MPMD-Size during step t is equal to $\varepsilon_t = \rho_t(\mathbf{u}_t)$ (recall that $\mathbf{u}_t$ is the only state in $\mathcal{K}^V$ where ρ_t incurs positive cost) if and only if $R_t \oplus M_t = \mathbf{1}$, which is equivalent to that $M_t = \mathbf{u}_t$ based on our setting of the MPMD-Size instance that $R_t \oplus \mathbf{u}_t = \mathbf{1}$. Thus, the service cost of $\mathcal{A}_{\mathrm{ms}}$ is equal to the delay cost of $\mathcal{A}_{\mathrm{size}}$. ◀

Now the reduction from MTS-Single to MPMD-Size has been established as the following:

▶ **Theorem 3.21.** *Let (V, d) be a metric, c_{ms} be the competitive ratio of MTS-Single on $(\mathcal{K}^V, D)$ and c_{size} be that of MPMD-Size on (V, d). Then, $c_{\mathrm{ms}} \leq c_{\mathrm{size}}$.*

Combined with Lemma 2.2, we can use c_{m}, the competitive ratio of the general MTS problem, to provide lower bounds for the MPMD-Size problem. Therefore, we obtain the following corollary when we apply the universal lower bound of $\Omega(N)$ for deterministic algorithms from [10] and that of $\Omega(\log N)$ for randomized algorithms from [11] of MTS on every N-point metric.

▶ **Corollary 3.22.** *No deterministic algorithm can achieve competitive ratio better than $\Omega(2^n)$ and no randomized algorithm can achieve competitive ratio better than $\Omega(n)$ for MPMD-Size on any n-point metric.*

This gives the lower bounds in Theorems 1.1 and 1.2.

4 MPMD-Convex on Uniform Metrics

In this section, we study another form of delay functions that does not fall in the category of size-based delays. We use the continuous time model to study the online matching with uniform delays, where the same monotone continuous delay function $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ (we further assume that $f(0) = 0$ without loss of generality) applies to every request r in the following way: suppose the arrival time of a request r is $a(r)$ and t is the time when it is matched by the algorithm, then the total delay cost incurred by request r is $f(t - a(r))$. In this section, we still use $\ell(r)$ to denote the location of request r in the metric space, and the connection cost of matching two requests r_1 and r_2 at locations $\ell(r_1)$ and $\ell(r_2)$ respectively is still $d(\ell(r_1), \ell(r_2))$. The total cost of a solution consists of its total connection cost of matching request into pairs, and the sum of delay cost incurred by every request.

We study monotone convex polynomial delay functions f such that $f'(0) > 0$ on n-point uniform metrics where the distance between any two different points is $2\delta > 0$. Equivalently, such a metric can be interpreted as a star metric where leaf nodes are points in the metric space, the distance between every leaf and the root is δ, and the distance between two leaves is the length of the shortest path between them, which is 2δ. The previous study by Liu et al. [20] gives a tight $\Theta(n)$ deterministic bound for a class of convex delay functions (including all monotone polynomials) such that $f(0) = f'(0) = 0$, and our study complements the landscape by the following result for functions that $f'(0) > 0$:

▶ **Theorem 4.1.** *For any fixed convex monotone polynomial delay f such that $f'(0) > 0$, there is a constant-competitive deterministic algorithm for MPMD under delay function f on all n-point uniform metrics with inter-point distance δ for constant δ.*

The contrast of this result compared with the $\Omega(n)$ deterministic lower bound of the case of continuous non-decreasing unbounded delays satisfying that $f(0) = f'(0) = 0$ demonstrates that even the local property of the delay function at time $t = 0$ significantly affects the solution and competitiveness of the online MPMD problem. Different from the linear delay case, here the distance 2δ between different points in the metric space also affects the competitive ratio: our algorithm is constant competitive under a fixed function f when δ is considered as a constant (the same as the study by Liu et al. [20]), while the competitive ratio can also be regarded as a function of δ, which provides another parameter that might be interesting to study for convex delays.

4.1 Simpler Polynomials

Without loss of generality, we study f in the form $f(t) = \frac{1}{k}t^k + t$ for $k \geq 2$. To see why we can make this assumption, suppose that we have another polynomial $h(t) = \sum_{i=1}^{k} a_i t^i$ that is monotone on $\mathbb{R}_{\geq 0}$, such that $a_k > 0, a_1 > 0$. [3] The following claim proves that h can be approximated by f within constant factors.

▷ Claim 4.2. There exist constants $A, B > 0$ such that $A \leq \frac{f(t)}{h(t)} \leq B$ for every $t \geq 0$.

Proof. Consider the rational function $\frac{f(t)}{h(t)}$. Since $h(0) = 0$ and h is monotone on $\mathbb{R}_{\geq 0}$, it holds that $h(t) > 0$ for every $t > 0$. As a result, $\frac{f(t)}{h(t)}$ is well-defined for every $t \in (0, +\infty)$.

Moreover, it holds that $\lim_{t\to 0^+} \frac{f(t)}{h(t)} = \frac{1}{a_1}$ according to L'Hôpital's rule and $\lim_{t\to+\infty} \frac{f(t)}{h(t)} = \frac{1}{ka_k}$, and both values are strictly bigger than zero based on our assumption that a_k and a_1 are both non-zero values and $h(t)$ is monotone on $\mathbb{R}_{\geq 0}$. Based on the definition of the

[3] We need $a_1 > 0$ to have $h'(0) > 0$.

limit, there exists $T > 0$ such that $0 < \frac{1}{2ka_k} < \frac{f(t)}{h(t)} < \frac{2}{ka_k}$ for every $t > T$. Because $\frac{f(t)}{h(t)}$ is a continuous positive function on the closed interval $[0, T]$ (define $\frac{f(0)}{h(0)} = \frac{1}{a_1}$), it has positive upper and lower bounds on $[0, T]$. Combined with the positive upper and lower bounds on $(T, +\infty)$, we conclude that $\frac{f(t)}{h(t)}$ has positive upper and lower bounds on $\mathbb{R}_{\geq 0}$. ◁

As a result, if there is a competitive algorithm for delay function f, then the same algorithm can be applied to delay function h and the lost factor of the competitive ratio is within a constant factor that only depends on A and B, independent of both n and δ.

If we apply this idea of comparing the costs of the same solution under different delay functions to comparing the convex delay function $f(t) = \frac{1}{k}t^k + t$ with the classical linear delay function $g(t) = t$ on the same request sequence, the only difference that can lead to different costs and different optimal solutions are delay functions f and g. In the following part, we focus on the linear delay function g, and prove that a competitive algorithm for g with certain properties also works well for f. Given an algorithm $\mathcal{A}$ and some delay function h, we use $\text{cost}_{\mathcal{A}}(h)$ to denote the cost of $\mathcal{A}$ when the delay function is set to h. As the total cost consists of connection cost and delay cost, we use $\text{cost}^d_{\mathcal{A}}(h)$ to denote its connection cost and $\text{cost}^t_{\mathcal{A}}(h)$ to denote its delay cost. Similarly for any solution SOL, we use $\text{SOL}(h)$ to denote its total cost under h, as well as $\text{SOL}^d(h)$ and $\text{SOL}^t(h)$ to denote its connection and delay costs respectively. These are the notations that we will use in computations in the next section.

4.2 T-Impatient Algorithms

For an online algorithm $\mathcal{A}$ and a request r, define $t_{\mathcal{A}}(r)$ to be the time that r is matched by $\mathcal{A}$.

▶ **Definition 4.3** (T-impatient algorithms). *An online algorithm $\mathcal{A}$ is* T-impatient *for a fixed value $T > 0$ if for every request r such that $t_{\mathcal{A}}(r) - a(r) > T$, r is the only pending request in the whole metric space at every moment throughout the whole time interval $[a(r) + T, t_{\mathcal{A}}(r))$, i.e. all previous requests have been matched and no new requests arrive during the interval.*

Intuitively, a T-impatient algorithm will only leave a request r unmatched for more than T time only if there is no other request it can match r to. Equivalently, if a request r has been pending for time T, then $\mathcal{A}$ is going to match r with another request immediately as long as there is such a request available. The T-impatient property guarantees a lower bound of the optimal solution during the period when there is a request pending for longer than T. On the other hand, when no request is pending for longer than T, the delay increment can be approximated by the linear delay up to a constant factor as long as $f'(0) > 0$. These two properties guarantee the competitiveness of a T-impatient algorithm under convex delays.

The next lemma is the key to guaranteeing that the cost of a T-impatient algorithm during the period when there is a request that has been pending beyond time T is within a constant factor of the cost incurred by the optimal solution during the same period.

▶ **Lemma 4.4.** *Given $f(t) = \frac{1}{k}t^k + t$ and a constant $T > 0$, $\frac{f(x)-f(T)}{f(x-T)} < 2^k(T^{k-1} + 1)$ for every $x > T$.*

We now formally state the reduction from convex delay $f(t) = \frac{1}{k}t^k + t$ to linear delay $g(t) = t$ under any T-impatient algorithm.

▶ **Lemma 4.5.** *On every n-point uniform metric, if a T-impatient algorithm $\mathcal{A}$ is γ-competitive ($\gamma \geq 1$) under delay function $g(t) = t$, then $\mathcal{A}$ is $2^{k+1}(T^{k-1} + 1)\gamma$-competitive under delay function $f(t) = \frac{1}{k}t^k + t$.*

The proofs of Lemmas 4.4 and 4.5 can be found in the full version.

With this reduction, now the goal is to find an T-impatient algorithm for some constant T independent of n that achieves competitive ratio γ under delay function $g(t) = t$, where γ is a constant independent of n as well.

▶ Remark 4.6. At this stage, it is natural to ask whether the previous algorithm that works well for the linear delay on a tree from [2] (as uniform metrics are a special class of trees with height 1 when we view the graph as a star) satisfies the T-impatient property. Unfortunately, this is not true and we do have to propose a new algorithm to be applied to convex delays. See the full version for a brief discussion on how their algorithm runs on a uniform star metric, and why it does not satisfy the T-impatient property for any constant T even when δ is a constant.

4.3 Constructing a T-Impatient Algorithm

In this section, we propose a T-impatient algorithm and prove that it is competitive under linear delay $g(t) = t$, and thus a competitive algorithm under $f(t) = \frac{1}{k}t^k + t$ according to Lemma 4.5.

High-level description. Recall that a uniform metric can be regarded as a star where each leaf corresponds to a point in the metric space, and the distance between each leaf and the root is δ. We assign a counter to every vertex of the star graph, i.e. all leaves and the root.

During the execution of our algorithm (Algorithm 2), each request can be in one of three states: LEAF, ROOT and READY. A request r is in LEAF if and only if it is contributing to the increase of the leaf counter $z_{\ell(r)}$ ($\ell(r)$ is the location of r), in ROOT if it is neither in LEAF nor in READY. If we want to view the dynamic of a request moving between states, then after a request r arrives, initially it is in LEAF until $z_{\ell(r)}$ is saturated (if $z_{\ell(r)}$ is already saturated at arrival time $a(r)$, we can still say that the time length when r is in LEAF is zero). Once $z_{\ell(r)}$ is saturated, r transits to ROOT, and will only transit from ROOT to READY if the condition of either Case 1 or Case 2 is satisfied. The state transition diagram LEAF $\to$ ROOT $\to$ READY is acyclic, which means that once a request r transits to the next state, there is no way back to the previous state. Based on the fact that the counter $z_{\ell(r)}$ is saturated is equivalent to that r is in ROOT or READY for a pending request r, the conditions to trigger a match between two requests at different points can also be expressed as: either both are in ROOT, or one is in READY.

Note that the priority rule guarantees that if one request is in READY, then it should be matched immediately as long as there is another pending request, which is vital in guaranteeing the T-impatient property.

▷ Claim 4.7. ALG is 4δ-impatient.

Proof. It suffices to show that a request r with arrival time $a(r)$ transits to state READY no later than $a(r) + 4\delta$. It takes time at most δ to transit from LEAF to ROOT and at most 3δ (the sum of Cases 1 and 2) to transit from ROOT to READY after the arrival of a request if it stays unmatched. ◁

Overview of analysis. Our analysis is similar to that in [2, 1] in the sense that we analyze counters associated with different points separately, where the time axis can be divided into phases for each counter according to when it is saturated, and we compare the algorithm with another given solution based on *alignment status* (to be defined later). The biggest difference is that their algorithms only match two requests on a uniform metric when both

Algorithm 2 A Deterministic Algorithm for MPMD on n-point uniform metrics with linear delay $g(t) = t$.

Initialize: For each leaf vertex v, $z_v \leftarrow 0$. For the root vertex u, $z_u \leftarrow 0$.

At every step t:

While there is a request r pending at $\ell(r)$:

1. if $z_{\ell(r)} < \delta$, then r contributes to the increase of the counter $z_{\ell(r)}$ with unit rate and r is in the state LEAF, otherwise if $z_{\ell(r)} = \delta$ then r is in ROOT until it transits to READY;
2. if $z_{\ell(r)} = \delta$ and r is the only pending request, then then r contributes to the increase of the root counter z_u;
3. if the single request r has contributed the total value δ to counter z_u, then r transits from ROOT to READY (Case 1);
4. if the total amount of time when r is in ROOT while not contributing to the increase of z_u accumulates to 2δ, then r transits from ROOT to READY (Case 2).

The conditions that trigger a match and counter resetting:

If there are two pending requests r_1 and r_2 such that $\ell(r_1) = \ell(r_2)$, match r_1 and r_2.

If there are two pending requests r_1 and r_2 such that $z_{\ell(r_1)} = z_{\ell(r_2)} = \delta$, match r_1 and r_2 and reset both $z_{\ell(r_1)}$ and $z_{\ell(r_2)}$ to 0.

If there are two pending requests r_1 and r_2 such that r_1 is in READY, match r_1 and r_2 and only reset $z_{\ell(r_1)}$ to 0.

Priority rule: If multiple matches can occur at the same time, then the request that transited to state ROOT at the earliest time (as long as it exists) is prioritized.

counters associated with these two requests are saturated, while our algorithm has another case that a request in READY is matched with another request immediately when it appears without waiting for another request's counter to also be saturated. This is the key property that makes Algorithm 2 satisfy the desired T-impatient property. To deal with this case for the cost analysis, we prove that the total cost incurred by matches of this case provides a lower bound of the optimal cost as well, hence proving that our new algorithm is also competitive.

Throughout the analysis, we use ALG to denote both the algorithm that we propose and its cost, and SOL to denote a fixed offline solution and its cost that we are comparing ALG with, and the costs are always incurred under delay function g. The superscripts d and t that represent connection and delay costs respectively work in the same way as before.

We first note that our algorithm is smart (defined in Section 3.1) as well, which will be used for the alignment status analysis that we are going to introduce.

▶ **Observation 4.8.** *There is at most one request pending at the same point in the metric space at every moment except when a match occurs.*

We can also assume that Observation 4.8 holds for every offline solution SOL, because linear delay is a special case of the size-based delay and Lemma 3.1 holds for this case as well. Our delay function is continuous, which means that the instantaneous delay increment at the discrete moments when a match occurs is negligible. As a result, if we want to compare the status of pending requests under ALG with another solution SOL, at every moment there are four cases at every point $v \in V$: there is one request pending at v under both ALG and SOL, there is no request pending at v under both ALG and SOL, there is one request pending at v under ALG but no request pending at v under SOL, there is one request pending at v under SOL but no request pending at v under ALG. We say that a point $v \in V$ is *aligned* if it is one of the first two cases among the four, otherwise v is *misaligned*.

▶ **Observation 4.9.** *The alignment status of a point $v \in V$ changes if and only if exactly one of* ALG *and* SOL *matches v with another point.*

We say that a request r *carries* v if ALG makes a match between r that is in READY and a request at $v \neq \ell(r)$. For such a request r, let $t_1(r)$ be the moment when it starts to be in ROOT and $t_2(r)$ be the moment when it starts to be in READY.

▷ Claim 4.10. Suppose that two different requests r and r' both have reached READY status under Algorithm 2, then $[t_1(r), t_2(r)) \cap [t_1(r'), t_2(r')) = \varnothing$.

Proof. Without loss of generality, assume that $t_1(r) \leq t_1(r')$. At the time $t_1(r')$ when r' reaches ROOT, if $t_2(r) > t_1(r')$, i.e., r is still unmatched in ROOT status, then r and r' should have been matched immediately at $t_1(r')$ if no other request matches either of them, a contradiction to that r, r' both have reached READY at some later time. ◁

▶ **Lemma 4.11.** *Suppose that a request r carries some point, then* SOL $|_{[t_1(r),t_2(r))} \geq \delta$.

Proof. We discuss different cases by how r transits from states ROOT to READY. By the design of ALG, now we cover cases 1 and 2 separately.

Case 1. Every moment when r is contributing to the increase of the root counter z_u, r is the only request that is pending, which means that the total number of arrivals of requests in the input sequence so far is an odd number. Therefore, under any solution SOL, there is at least one request pending and hence the delay is increasing with rate at least 1 at every moment when z_u is increasing. By the condition of Case 1 that r has contributed δ to the increase of z_u, the delay cost of SOL during this period must be at least δ as well.

Case 2. It is easy to see that ALG does not match any two requests at different points during $(t_1(r), t_2(r))$, since every such match must involve another request that has just reached state ROOT, while the priority rule would have matched r instead. If SOL makes a match between two requests at different points, then cost of at least 2δ is incurred.

Otherwise, by Observation 4.9, the alignment of every point stays unchanged throughout $(t_1(r), t_2(r))$. Let $\mathbf{t} = t_2(r) - t_1(r)$ denote the length of this interval, then $\mathbf{t} \geq 2\delta$ by the condition of Case 2 that r transits into READY. For all other points in the metric than $\ell(r)$ denoted by $v_1, \ldots, v_{n-1}$ respectively, let $x_1, \ldots, x_{n-1}$ denote the increased counter value of $v_1, \ldots, v_{n-1}$ during this period respectively, then $x_i \leq \delta$ for every $1 \leq i \leq n-1$, otherwise some counter z_{v_i} should have been saturated since no counter is reset in this period and r should have been matched according to the priority rule. Moreover, $x_1 + \ldots + x_{n-1} \geq 2\delta$, since the total time when there is at least one counter increasing among $v_1, \ldots, v_{n-1}$ has reached 2δ for r to transit into READY. By the definition of the alignment status, for every v_i, either it is aligned and therefore there is a request pending at it throughout the total time x_i when z_{v_i} is increasing, or it is misaligned and therefore there is a request pending at it throughout the total time $\mathbf{t} - x_i$ when z_{v_i} is not increasing. In either way, the delay cost incurred by a request pending at v_i is at least $\min\{x_i, \mathbf{t} - x_i\}$, which is equal to x_i by the fact that $x_i \leq \delta$ and $\mathbf{t} \geq 2\delta$. As a result, the total delay time incurred at all points in $v_1, \ldots, v_{n-1}$ is at least $x_1 + \ldots + x_{n-1} \geq 2\delta$.

Therefore, the cost incurred by SOL is at least δ during $[t_1(r), t_2(r))$ for both cases. ◀

The following corollary is a direct result of combining Claim 4.10 and Lemma 4.11.

▶ **Corollary 4.12.** *Let $R = \{r \mid r$ carries some point during the execution of* ALG$\}$*, then*

$$\text{SOL} \geq \sum_{r \in R} \delta |R|.$$

Now we start to analyze the cost of ALG under the linear delay $g(t) = t$ based on the increments of all leaf counters z_v's and z_u. For each leaf counter z_v, let y_v denote the total increments of the value of z_v throughout the execution of ALG without any reset. Similarly, let y_u denote the final value of the root counter z_u since the value of z_u is never reset to zero. We refer the reader to the full version of our paper for the proofs of the remaining lemmas in this section.

▶ **Lemma 4.13.** $\mathrm{ALG}^t \le 2\sum_{leaf\ v} y_v + y_u$.

▶ **Lemma 4.14.** $\mathrm{ALG}^d \le 2\sum_{leaf\ v} y_v$.

Now we need to relate the final value y_u of the root counter and y_v's of leaf counters to the cost of any SOL. Given a solution SOL, for every leaf v, let x_v denote the delay cost incurred by requests at v, x'_v denote the connection cost incurred by matching v with another point (i.e. $x'_v = k\delta$ if v is matched with another point for k times by SOL, and the total connection cost is just the sum among all leaf points). We introduce an additional term based on ALG rather than SOL, which is b_v, to denote the number of times that v is *carried* by another request under the execution of ALG.

▶ **Lemma 4.15.** $y_u \le \mathrm{SOL}^t$.

▶ **Lemma 4.16.** $y_v \le 2x_v + x'_v + \delta \cdot b_v$ *for every leaf* v.

▶ **Theorem 4.17.** *Algorithm 2 achieves competitive ratio* 13 *under delay function* $g(t) = t$ *and* $13 \cdot 2^{k+1}((4\delta)^{k-1} + 1)$ *under delay function* $f(t) = \frac{1}{k}t^k + t$ *for every* $k \ge 2$.

Proof. From Lemmas 4.13 and 4.14, the algorithmic cost $\mathrm{ALG} = \mathrm{ALG}^t + \mathrm{ALG}^d$ is at most $4\sum_{\mathrm{leaf}\ v} y_v + y_u$. By Lemmas 4.15 and 4.16, this is at most $\mathrm{SOL}^t + \sum_{\mathrm{leaf}\ v}(8x_v + 4x'_v + 4\delta \cdot b_v) = 9 \cdot \mathrm{SOL}^t + 4 \cdot \mathrm{SOL}^d + 4\delta \cdot \sum_{\mathrm{leaf}\ v} b_v$, where b_v is the number of times that v is carried by another request. By Corollary 4.12, $\delta \cdot \sum_{\mathrm{leaf}\ v} b_v \le \mathrm{SOL}$. All combined together, we have $\mathrm{ALG} \le 13 \cdot \mathrm{SOL}$, and this is the competitive ratio that ALG achieves under the linear delay function. According to Lemma 4.5, it achieves competitive ratio $13 \cdot 2^{k+1}((4\delta)^{k-1} + t)$ under the delay function $f(t) = \frac{1}{k}t^k + 1$ for $k > 1$. ◀

▶ Remark 4.18. In this section, we use functions of the form $f(t) = \frac{1}{k}t^k + t$ for every $k > 1$ as representatives as all monotone convex polynomials, and show that the competitive ratio can be preserved up to a constant factor independent of n and δ because of the approximation properties as Claim 4.2. In fact, the $O(1)$-competitive ratio can be achieved by the same algorithm for a broader class of non-negative, non-decreasing convex functions where $f'(0) > 0$, as long as they hold similar properties to Lemma 4.4 that $\frac{f(x)-f(T)}{f(x-T)}$ is upper bounded by a positive value depending on T when $x \to \infty$ for any $T > 0$. This covers every non-negative, non-decreasing and smooth convex function f that $f'(0) > 0$ whose leading term is $\Theta(x^\alpha)$ for any $\alpha > 1$, as well as $\Theta(e^x)$. However, exceptions do exist. For example, $f(x) = e^{x^2} - 1$ does not satisfy this property.

5 Discussions and Open Problems

In this paper, we improved both upper and lower bounds of MPMD-Size on every n-point metric space. We show that the problem is equivalent to MTS on another 2^{n-1}-point metric space, and the results are consistent with the state-of-the-art results of MTS. The deterministic competitive ratio is settled up to a constant as the $\Theta(N)$ bound of MTS holds for every N-point metric. However, there is still a gap between the upper and lower

bounds for randomized algorithms. This is because the randomized competitive ratio of MTS depends on metric structures: as the randomized competitive ratio of $O(\log N)$ can be achieved on some N-point metrics like uniform metrics [10] and weighted stars [12], there also exist N-point metrics where the lower bound of $\Omega(\log^2 N)$ holds [11]. As a result, to make further improvements, we need to study the structure of the metric space $(\mathcal{K}^V, D)$. Given a metric space (V, d), the definition of $(\mathcal{K}^V, D)$ is naturally from the Boolean status of every point in v, with some similarities to the earthmover distance, yet this structure has not been well studied, and may be of independent interest.

Our study of MPMD-Convex on uniform metrics covers the case $f'(0) > 0$ that has not been considered before, and very surprisingly, such local property at $t = 0$ totally changes the competitiveness compared with the case $f'(0) = 0$. As the majority of smooth convex delay functions have been covered on uniform metrics, the natural open problem is to design an algorithm for convex delays on more general metrics. There are still some open problems on uniform metrics as well: when we consider randomization, there is only a lower bound of $\Omega(\log n)$ for unbounded, non-decreasing and continuous functions that $f(0) = f'(0) = 0$, while it is open whether there is a randomized algorithm that can achieve competitive ratio of $O(\log n)$ in this case. As for the case $f'(0) > 0$, we note that although Algorithm 2 achieves constant competitive ratio when the distance between points is a fixed constant 2δ, the competitive ratio depends on δ and tends to infinity when δ is sufficiently large. This is different from the linear delay case where the scale of the metric does not affect the competitive ratio. As a result, it might be interesting to study whether it is possible to design an algorithm whose competitive ratio is also independent of δ.

Another natural research direction is to study the bipartite case of MPMD-Size and MPMD-Convex, where every request has either positive or negative polarity and only requests of opposite polarities can be matched. The online min-cost bipartite perfect matching with delay problem has been studied for linear [1] and concave delays [5], yet there is no result for convex or size-based delay. While the proofs and results by Deryckere and Umboh [14] can be easily adapted for the bipartite matching problem by treating every even-sized subset whose sum of polarities is zero as an MTS state, our method for MPMD-Size does not easily extend to the bipartite case as it is not sufficient to express the states with Boolean vectors because multiple requests of the same polarity can accumulate at the same location.

References

1 Itai Ashlagi, Yossi Azar, Moses Charikar, Ashish Chiplunkar, Ofir Geri, Haim Kaplan, Rahul Makhijani, Yuyi Wang, and Roger Wattenhofer. Min-cost bipartite perfect matching with delays. In Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, volume 81 of *LIPIcs*, pages 1:1–1:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.1.

2 Yossi Azar, Ashish Chiplunkar, and Haim Kaplan. Polylogarithmic bounds on the competitiveness of min-cost perfect matching with delays. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1051–1061. SIAM, 2017. doi:10.1137/1.9781611974782.67.

3 Yossi Azar, Ashish Chiplunkar, Shay Kutten, and Noam Touitou. Set cover with delay - clairvoyance is not required. In Fabrizio Grandoni, Grzegorz Herman, and Peter Sanders, editors, *28th Annual European Symposium on Algorithms, ESA 2020, Pisa, Italy (Virtual Conference), September 7-9, 2020*, volume 173 of *LIPIcs*, pages 8:1–8:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ESA.2020.8.

4 Yossi Azar and Amit Jacob Fanani. Deterministic min-cost matching with delays. *Theory Comput. Syst.*, 64(4):572–592, 2020. doi:10.1007/S00224-019-09963-7.

5 Yossi Azar, Runtian Ren, and Danny Vainstein. The min-cost matching with concave delays problem. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 301–320. SIAM, 2021. doi:10.1137/1.9781611976465.20.

6 Nikhil Bansal, Niv Buchbinder, Aleksander Madry, and Joseph Naor. A polylogarithmic-competitive algorithm for the k-server problem. *J. ACM*, 62(5):40:1–40:49, 2015. doi:10.1145/2783434.

7 Sujoy Bhore, Michał Pawłowski, and Seeun William Umboh. Online tcp acknowledgment under general delays. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2026*, 2026. To appear. Full version available as arXiv:2604.13428 [cs.DS]. doi:10.48550/arXiv.2604.13428.

8 Marcin Bienkowski, Artur Kraska, Hsiang-Hsuan Liu, and Pawel Schmidt. A primal-dual online deterministic algorithm for matching with delays. In Leah Epstein and Thomas Erlebach, editors, *Approximation and Online Algorithms - 16th International Workshop, WAOA 2018, Helsinki, Finland, August 23-24, 2018, Revised Selected Papers*, volume 11312 of *Lecture Notes in Computer Science*, pages 51–68. Springer, 2018. doi:10.1007/978-3-030-04693-4_4.

9 Marcin Bienkowski, Artur Kraska, and Pawel Schmidt. A match in time saves nine: Deterministic online matching with delays. In Roberto Solis-Oba and Rudolf Fleischer, editors, *Approximation and Online Algorithms - 15th International Workshop, WAOA 2017, Vienna, Austria, September 7-8, 2017, Revised Selected Papers*, volume 10787 of *Lecture Notes in Computer Science*, pages 132–146. Springer, 2017. doi:10.1007/978-3-319-89441-6_11.

10 Allan Borodin, Nathan Linial, and Michael E. Saks. An optimal online algorithm for metrical task systems. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 373–382. ACM, 1987. doi:10.1145/28395.28435.

11 Sébastien Bubeck, Christian Coester, and Yuval Rabani. The randomized k-server conjecture is false! In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 581–594. ACM, 2023. doi:10.1145/3564246.3585132.

12 Sébastien Bubeck, Michael B. Cohen, James R. Lee, and Yin Tat Lee. Metrical task systems on trees via mirror descent and unfair gluing. *SIAM J. Comput.*, 50(3):909–923, 2021. doi:10.1137/19M1237879.

13 Ryder Chen, Jahanvi Khatkar, and Seeun William Umboh. Online weighted cardinality joint replenishment problem with delay. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, Paris, France, July 4-8, 2022*, volume 229 of *LIPIcs*, pages 40:1–40:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ICALP.2022.40.

14 Lindsey Deryckere and Seeun William Umboh. Online matching with set and concave delays. In Nicole Megow and Adam D. Smith, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2023, Atlanta, Georgia, USA, September 11-13, 2023*, volume 275 of *LIPIcs*, pages 17:1–17:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.APPROX/RANDOM.2023.17.

15 Marc Dufay and Roger Wattenhofer. A deterministic polylogarithmic competitive algorithm for matching with delays. In Kasper Green Larsen and Barna Saha, editors, *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2026, Vancouver, BC, Canada, January 11-14, 2026*, pages 3936–3964. SIAM, 2026. doi:10.1137/1.9781611978971.144.

16 Yuval Emek, Shay Kutten, and Roger Wattenhofer. Online matching: haste makes waste! In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 333–344. ACM, 2016. doi:10.1145/2897518.2897557.

17 Leah Epstein. On bin packing with clustering and bin packing with delays. *Discret. Optim.*, 41:100647, 2021. doi:10.1016/J.DISOPT.2021.100647.

18 Tomer Ezra, Stefano Leonardi, Michal Pawlowski, Matteo Russo, and Seeun William Umboh. Universal optimization for non-clairvoyant subadditive joint replenishment. In Amit Kumar and Noga Ron-Zewi, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2024, London School of Economics, London, UK, August 28-30, 2024*, LIPIcs, pages 12:1–12:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.APPROX/RANDOM.2024.12.

19 Ngoc Mai Le, Seeun William Umboh, and Ningyuan Xie. The power of clairvoyance for multi-level aggregation and set cover with delay. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 1594–1610. SIAM, 2023. doi:10.1137/1.9781611977554.CH59.

20 Xingwu Liu, Zhida Pan, Yuyi Wang, and Roger Wattenhofer. Impatient online matching. In Wen-Lian Hsu, Der-Tsai Lee, and Chung-Shou Liao, editors, *29th International Symposium on Algorithms and Computation, ISAAC 2018, December 16-19, 2018, Jiaoxi, Yilan, Taiwan*, volume 123 of *LIPIcs*, pages 62:1–62:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ISAAC.2018.62.

Online TCP Acknowledgment Under General Delays

Sujoy Bhore
Department of Computer Science & Engineering, Indian Institute of Technology Bombay, India

Michał Pawłowski
Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Poland
Department of Computer, Control and Management Engineering,
Sapienza University of Rome, Italy
IDEAS NCBR, Warsaw, Poland

Seeun William Umboh
School of Computing and Information Systems, The University of Melbourne, Australia
ARC Training Centre in Optimisation Technologies, Integrated Methodologies, and Applications (OPTIMA), Melbourne, Australia

Abstract

In a seminal work, Dooly, Goldman, and Scott (STOC 1998; JACM 2001) introduced the classic *Online TCP Acknowledgment* problem. In this problem, a sequence of n packets arrives over time, and the objective is to minimize both the number of acknowledgments sent and the total delay experienced by the packets. They showed that a natural greedy algorithm, which acknowledges when the delay of pending packets equals the acknowledgment cost, is 2-competitive.

Online TCP Acknowledgment is the canonical online problem with delay, capturing the fundamental tradeoff between reducing service cost through batching and the delay incurred by pending requests. Prior work has largely focused on richer service-cost models, e.g., Joint Replenishment and Multi-Level Aggregation. However, other than the work of Albers and Bals (SODA 2003), which studies maximum delay and closely related objectives, not much is known about general delay costs beyond the sum of delay costs of requests.

In this work, we study Online TCP Acknowledgment under two generalized delay-cost models that we call *batch-aware* and *batch-oblivious*. In the batch-aware model, each batch incurs a delay cost that depends on the packet delays within that batch. For the max-over-batches objective, which generalizes Albers and Bals, we show that greedy remains 2-competitive for every monotone batch-delay function. For the sum-over-batches objective, the picture changes sharply: greedy is $\Omega(n)$-competitive, and the optimal deterministic competitive ratio is $\Theta(\log n)$. Our matching upper bound requires only the minimal assumption that the batch delay function is monotone.

In the batch-oblivious model, the delay cost is a function of the global packet-delay vector. We show that greedy is 2-competitive for continuous submodular delay costs, and more generally under a weaker zero-coordinate diminishing-marginals condition. This yields 2-competitive algorithms for ℓ_p norms, Top-k norms, and ordered norms. Using the submodular-norm approximation of Patton, Russo, and Singla, we also obtain an $O(\log n)$-competitive algorithm for arbitrary symmetric norms.

2012 ACM Subject Classification Theory of computation → Online algorithms

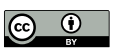
Keywords and phrases Online Algorithms, TCP Acknowledgment, General Delay Functions

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.26

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2604.13428

Funding *Sujoy Bhore*: Work supported in part by ANRF ARG-MATRICS, Grant 002465.
Michał Pawłowski: Supported by the Polish National Science Center grant no 2024/53/N/ST6/04119.
Seeun William Umboh: Supported by the Australian Government through the Australian Research Council DP240101353.



© Sujoy Bhore, Michał Pawłowski, and Seeun William Umboh;
licensed under Creative Commons License CC-BY 4.0

1 Introduction

The *Online TCP Acknowledgment* problem, introduced by Dooly, Goldman, and Scott in their seminal STOC 1998 paper [28], arose from critical challenges in computer networking. Specifically, it addresses the dynamic management of acknowledgment delays within the Transmission Control Protocol (TCP), a cornerstone of the Internet's architecture (see [4, 24, 25]). The TCP protocol requires the recipient of a stream of packets to acknowledge every packet. On the other hand, excessive delay can disrupt TCP's congestion control mechanisms. Consequently, limiting the latency between a packet's arrival and its acknowledgment is vital for maintaining network performance and protocol stability.

In the Online TCP Acknowledgment problem – the TCP problem henceforth – a sequence of packets arrive online over time. At each time, we can transmit an acknowledgment to acknowledge all currently pending packets at a cost of 1. Each packet accrues a delay cost until it is acknowledged. More formally, the TCP problem involves a sequence of packet arrival times $A = (a_1, \ldots, a_n)$, and the objective is to partition A into k contiguous subsequences $\sigma_1, \sigma_2, \ldots, \sigma_k$, for some k. The end of each subsequence corresponds to the time an acknowledgment is transmitted. Thus, each σ_i is a *batch* of packets that are acknowledged together by the acknowledgment sent at time t_i. The batches σ_i and the acknowledgment times t_i are required to satisfy the following constraints: each packet can only be acknowledged after its arrival, i.e. $t_i \geq a_j$ for every $j \in \sigma_i$; and every packet must be acknowledged, i.e. the final acknowledgment time $t_k \geq a_n$, the arrival time of the last packet. The tradeoff between the cost of acknowledgments and the cost of delaying acknowledgments is modeled with the objective function

$$k + \sum_{i=1}^{k} \text{delay}(\sigma_i, t_i). \tag{1}$$

The number of acknowledgments k is called the *acknowledgment cost* and $\sum_{i=1}^{k} \text{delay}(\sigma_i, t_i)$ the *delay cost*. We refer to $\text{delay}(\cdot, \cdot)$ as the *batch delay cost function*.

TCP with Additive Delay Cost. The most classic and well-studied setting is when the delay function $\text{delay}(\sigma_i, t_i) = \sum_{j \in \sigma_i} (t_i - a_j)$, i.e. the delay cost is simply the total waiting time between each packet's arrival time and its acknowledgment time. The competitiveness of this problem has been completely characterized. For deterministic algorithms, Dooly, Goldman, and Scott [28, 29] proved that the following natural (in hindsight) Greedy algorithm achieves a competitive ratio of 2: acknowledge whenever the delay cost has increased by the cost of an acknowledgment. This is also the optimal deterministic competitive ratio as TCP with additive delay generalizes the well-known Ski Rental problem for which a lower bound of 2 is known [40]. Subsequently, Seiden [48] established a lower bound of $e/(e-1) \approx 1.58$ on the competitive ratio of randomized algorithms; a matching upper bound was shown by Karlin, Kenyon, and Randall [39] shortly after. These algorithms also work for slightly more general delay costs: each request j incurs a delay cost c_j that is a non-decreasing function of its waiting time, and the overall delay cost is $\sum_{j=1}^{n} c_j$. We call this more general problem *TCP with additive delay*, since the delay cost is the sum of the delay costs of the requests.

Beyond TCP. TCP with additive delay is the simplest problem in the class of *online problems with delay*, which captures trade-offs between batching requests to reduce service costs and minimize delays. An online problem with delay involves a sequence of requests that arrive online and a *service cost function* g which is a function on the set of requests; the

goal is to partition the requests into (possibly non-contiguous) batches and a service time t_i for each batch σ_i so as to minimize the objective function $\sum_i g(\sigma_i) + \text{delay}(\sigma_i, t_i)$. The sum $\sum_i g(\sigma_i)$ is called the *service cost*.

Most prior work on online problems with delay has focused on objective functions with more general service costs, with the overall delay cost computed as the sum of the delay costs of requests. These objective functions capture various types of economies of scale arising in varied practical domains such as scheduling, supply chain management, and logistics. The ones most relevant to our work are: joint replenishment [21, 16, 23, 37, 45, 49, 9, 27], multi-level aggregation [19, 20, 14, 15, 42], network design with delay [11, 12, 51], set cover [7, 50, 33], set aggregation [22], online service with delay [8, 18, 41, 35, 36, 52, 53], and matching [31, 6, 5, 17, 32, 10, 26, 38, 43, 30, 34].

Beyond Additive Delay Cost. In many of the above practical domains, it is natural to consider non-additive delay costs, where the penalty incurred by delaying requests is not simply the sum of their waiting times or delay costs. For example, one may wish to control tail latency by ensuring that no request waits too long, or, more generally, by penalizing the largest few waiting times. In settings with service-level agreements, another natural objective is to limit the number of requests that experience positive delay or exceed a prescribed delay threshold. Such objectives capture global quality-of-service or incident-based costs that are not well represented by additive delays.

The main goal of our work is to initiate a systematic study of online problems with delay beyond additive delay costs, starting with the setting that has the simplest service cost function: online TCP Acknowledgment. We investigate the following broad questions.

▶ **Question 1.** *For what families of natural delay costs can we obtain a good competitive ratio for the online TCP acknowledgment problem?*

As Greedy is heavily used as a subroutine in many algorithms for online problems with delay to determine service times (e.g. in [12, 23]), a good understanding of the power of Greedy can help us extend those algorithms to work with non-additive delay costs.

▶ **Question 2.** *For what families of natural delay costs is Greedy a good algorithm for the online TCP Acknowledgment problem?*

To the best of our knowledge, the only previous works on online problems with delay that go beyond additive delay costs are those below.

General Batch Delay Cost. Dooly, Goldman, and Scott [28, 29] considered a far-reaching generalization of the objective in Equation (1) where the batch delay cost function $\text{delay}(\cdot, \cdot)$ is only assumed to be monotone non-decreasing: if σ is a subsequence of σ' and $t \leq t'$, then $\text{delay}(\sigma, t) \leq \text{delay}(\sigma', t)$ and $\text{delay}(\sigma, t) \leq \text{delay}(\sigma, t')$. Equivalently, the marginal delay cost at each timestep depends only on the set of currently pending requests. These general batch-delay cost functions can be used to model not only the goal of avoiding requests waiting too long but also the goal of avoiding too many pending requests at any time. The former can be modeled using ℓ_p-norms, while the latter can be modeled by taking $\text{delay}(\sigma_i, t_i)$ to be a function of $|\sigma_i|$.[1] Dooly, Goldman, and Scott showed that Greedy is 2-competitive when $\text{delay}(\sigma_i, t_i) = \max_{j \in \sigma_i}(t_i - a_i)$, i.e. the batch delay cost is the maximum waiting time of

[1] This type of delay cost function was also studied under the name "size-based delay" by Deryckere and Umboh [26] and Gan, Sun, and Umboh [34] for online matching with delay.

the batch. They also claimed that Greedy is 2-competitive for monotone, nondecreasing batch-delay costs. However, we show that Greedy is actually $\Omega(n)$-competitive, and thus new algorithms are required.

Penalizing Long Delays. Later, Albers and Bals [2, 3] explored a structurally different objective function of the form

$$k + \max_{i=1}^{k} \max_{j \in \sigma_i} (t_i - a_j)^p$$

for some integer $p \geq 1$. When $p = 1$, the delay cost is simply the maximum waiting time of any packet. They gave tight upper and lower bounds on the competitive ratio of deterministic online algorithms: when $p = 1$, it is $\pi^2/6 \approx 1.644$; when $p > 1$, it is an expression that approaches 1.5 as p tends to infinity. For randomized algorithms, they showed lower bounds of $3/(3 - 2/e) \approx 1.324$ and $2/(2 - 1/e) \approx 1.225$, when $p = 1$ and when $p > 1$, respectively. A comprehensive overview of this area until 2003 can be found in Chapter 7.2 of the survey by Albers [1].

1.1 Our Contributions

In this work, we make significant progress towards answering the above questions. We consider three broad classes of delay costs that generalize those considered in previous works: *max-monotone*, *sum-monotone*, and *batch-oblivious*.

Max-Monotone Delay Cost. We begin by considering a generalization of the delay cost considered by Albers and Bals [3]. In the *max-monotone delay* setting, the objective function is

$$k + \max_{i=1}^{k} \text{delay}(\sigma_i, t_i) \tag{Max-Monotone}$$

for some monotone non-decreasing $\text{delay}(\cdot, \cdot)$. We show that Greedy is 2-competitive without any further restrictions on $\text{delay}(\cdot, \cdot)$ using a similar analysis as Greedy under additive delay costs.

▶ **Theorem 1.1.** *Greedy is a deterministic* 2*-competitive algorithm for TCP Acknowledgment with max-monotone delay cost.*

Sum-Monotone Delay Cost. Our main technical contributions are for the *sum-monotone delay* setting, where the objective is

$$k + \sum_{i=1}^{k} \text{delay}(\sigma_i, t_i) \tag{Sum-Monotone}$$

for some monotone non-decreasing $\text{delay}(\cdot, \cdot)$.

Intuitively, given the success of Greedy for max-monotone delay, and the apparent simplicity of the problem – that at any point in time, we only need to decide between two actions (to acknowledge or not) – one expects that Greedy is also good for sum-monotone. Indeed, Dooly, Goldman, and Scott [29, Theorem 10] claimed that the same analysis of Greedy under additive delays can be extended to sum-monotone delays. Surprisingly, we show that Greedy is actually terrible.

▶ **Theorem 1.2** (Informal; see Theorem 4.1). *Greedy is $\Omega(n)$-competitive for TCP Acknowledgment with sum-monotone delay cost.*

A further surprise is that there is no deterministic constant-competitive algorithm for sum-monotone delays. We give a lower bound of $\Omega(\log n)$ on the deterministic competitive ratio for sum-monotone delays via the Parking Permit problem – a generalization of Ski Rental – and a matching upper bound.

▶ **Theorem 1.3.** *The deterministic competitive ratio for TCP Acknowledgment with sum-monotone delay cost is $\Theta(\log n)$.*

Note that we measure the waiting time of a packet $j \in \sigma_i$ as $t_i - a_j$. However, all of our results also apply if the waiting time is transformed by any monotone, nondecreasing (possibly nonlinear) function before aggregation. Since all delay cost functions we consider are monotone non-decreasing, such transformations can equivalently be absorbed into the definition of the batch delay cost function $\text{delay}(\cdot, \cdot)$.

Batch-Oblivious Delay Cost. In the *batch-oblivious delay* setting, the delay cost is given by applying a monotone non-decreasing function f to the *delay vector* d of waiting times of each packet. More precisely, the j-th coordinate of the vector d is the waiting time of the j-th packet. To summarize, the batch-oblivious objective is

$$k + f(d). \tag{Batch-Oblivious}$$

Unlike max-monotone and sum-monotone delays, batch-oblivious delays are "global" objectives. For example, we can measure the ℓ largest waiting times among all requests by taking f to be the Top-ℓ norm;[2] note that this objective is not captured by sum-monotone and max-monotone delays.

In this setting, we show that a natural extension of Greedy is 2-competitive not only for ℓ_p norms and Top-ℓ norms, but also for a broader class of functions that are *continuous submodular*. The proof is much subtler than for additive delay.

▶ **Theorem 1.4.** *Greedy is a deterministic 2-competitive algorithm for TCP Acknowledgment with batch-oblivious continuous submodular delay cost.*

Continuous submodularity (Definition 5.1) generalizes the notion of submodularity for discrete functions to functions on real vectors [13]. Patton, Russo, and Singla [47] observed that ℓ_p norms and several other symmetric norms are continuous submodular. They also showed that any symmetric norm can be approximated by a continuous submodular norm up to a logarithmic factor. Using the results of [47], we get the following result.

▶ **Corollary 1.5.** *There is a deterministic 2-competitive algorithm for TCP Acknowledgment, where the delay cost is: an ℓ_p norm, a Top-ℓ norm, or an ordered norm. When the delay cost is a symmetric norm $\|\cdot\|$, then there is a deterministic $O(\log \rho)$-competitive algorithm where $\rho = \|(1, 1, \ldots, 1)\| \,/\, \|(1, 0, \ldots, 0)\| \leq n$. In particular, there is a deterministic $O(\log n)$-competitive algorithm when the delay cost is a monotone symmetric norm.*

Our final result shows that if we venture too far from norms to concave functions, then the problem becomes significantly more difficult.[3]

▶ **Theorem 1.6.** *Every deterministic algorithm is $\Omega(\sqrt{n})$-competitive for TCP Acknowledgment with batch-oblivious concave delay cost.*

[2] The Top-ℓ norm of a vector is the sum of its ℓ-largest entries.
[3] Recall that norms are convex.

1.2 Our Techniques

We first develop intuition for our algorithms by recalling the Greedy algorithm and its analysis for the classic additive delay costs. Then, we describe the modifications required to prove that Greedy is 2-competitive for max-monotone and continuous submodular delay costs. Finally, we give an overview of our algorithmic framework for sum-monotone delays, where greedy strategies are no longer competitive, and outline the phase-based approach that overcomes this barrier, together with the accompanying logarithmic lower bound.

Recap: Additive Delay. The Greedy algorithm can be easily described as follows: send an acknowledgment when the total increase in its delay cost since the last acknowledgment equals 1 – the cost of an acknowledgment. For additive delays, this is equivalent to sending an acknowledgment when the total waiting time of unacknowledged requests since the last acknowledgment reaches 1. Let us now analyze the competitive ratio of Greedy for additive delays. Suppose Greedy makes k acknowledgments at times $t_1 < \ldots < t_k$, with batches $\sigma_1, \ldots, \sigma_k$. The analysis consists of two main components:

1. By design, Greedy's acknowledgment cost equals its delay cost, so its cost is exactly $2k$.
2. The optimal solution pays at least 1 during each of the k intervals $I_j = [t_{j-1}, t_j)$ for $j \in [k]$, where $t_0 = 0$. This is because with additive delays, if the optimal solution does not send an acknowledgment within the time interval I_j, then the packets in batch σ_i contribute $\text{delay}(\sigma_i, t)$ to its delay cost.

While the first component still holds for any monotone delay cost, the second component breaks for the delay costs that we consider.

Max-Monotone Delay. For max-monotone delays, we get that $\text{delay}(\sigma_j, t_j) = j$ and if the optimal solution does not send an acknowledgment during interval I_j, then the packets of σ_j were acknowledged together at some time $t_j^* \geq t_j$, and so by monotonicity of $\text{delay}(\cdot, \cdot)$, the delay cost of the optimal solution is at least j. Thus, if the optimal solution makes $k^* < k$ acknowledgments, its delay cost is at least $k - k^*$, and so its total cost is at least k.

Sum-Monotone Delay. For sum-monotone delay costs, greedy strategies are not competitive. We show that for any constant threshold, there exists an instance where a greedy algorithm incurs a cost that is $\Omega(n)$ times larger than that of the offline optimum. This fundamental limitation stems from the additive nature of the delay cost: greedy algorithms react too locally, failing to recognize when it would be better to wait and serve a larger group of packets with a single acknowledgment. For instance, if the delay function is concave and grows steadily up to the acknowledgment cost, and if the packets are spaced far enough apart, greedy will serve each packet individually – incurring an acknowledgment cost each time – while the optimal solution serves all at once at the last packet arrival, paying the delay cost only once and saving on acknowledgments.

To overcome this, we use our offline dynamic programming solution to guide online decisions. The key idea is to monitor the cost of the optimal solution on the packets seen so far. We partition the execution of the algorithm into consecutive *services*, each of which aggregates packets and ends with an acknowledgment. At the start of a service, we examine the sequence of packets observed so far and identify the longest suffix for which the offline optimum issues a single acknowledgment. The cost of optimally serving this suffix determines a *budget* for the service. The algorithm then continues to accumulate packets and issues an acknowledgment as soon as the total delay cost exceeds this budget. Thus, the service budget is derived directly from the structure of an optimal offline solution on the currently observed packets.

To avoid repeatedly issuing acknowledgments with small accumulated delay costs, each budget service is followed by three *buffer services* whose budgets are larger by a constant factor than that of the preceding budget service. These buffer services separate consecutive budget services and later allow us to analyze their costs independently. Throughout the execution, the algorithm continues to monitor newly observed suffixes. During a budget service, the arrival of any longer critical suffix triggers an update of the service budget, whereas during a buffer service, such an update occurs only if the offline cost of a suffix increases by a constant factor. This controlled budget growth ensures that the number of services grows only logarithmically, implying that the total cost incurred by the algorithm is within an $O(\log n)$ factor of the offline optimum.

Lower Bound for Sum-Monotone Delay. Observe that if the batch delay cost depends only on the maximum waiting time of the requests in the batch, then the offline problem essentially reduces to an interval covering problem, where we seek to cover the arrival times with intervals whose cost is a function of the interval length. Thus, it makes sense to consider the Parking Permit Problem [44], a generalization of the classic Ski Rental problem. In this problem, we are given different permit types and for each permit type k, its cost C_k and duration D_k. Purchasing permit k at time t covers the time up to $t + D_k$. Online, a sequence of requests over time, and for each requested time t, we need to ensure that it is covered by a purchased permit. A deterministic lower bound of $\Omega(\log n)$ is known for this problem [46].

Offline, it is straightforward to reduce Parking Permit to our problem, since both problems are interval covering problems in which the cost of an interval depends only on its length. However, it is unclear how to do the reduction online. Intuitively, we would like to use an algorithm for our problem to solve Parking Permit as follows: when a request arrives for Parking Permit at time t, we give the same request to the TCP Acknowledgment algorithm, see how long it waits before acknowledging the request, and buy a permit of the corresponding length. Unfortunately, at the time the request arrives, the TCP Acknowledgment algorithm does not need to commit to when it will acknowledge the request. Indeed, if the next request arrives immediately after, it may be acknowledged at a different time than if it arrives much later. We leave the possibility of an online reduction from Parking Permit to our problem as an open, tantalizing problem.

Fortunately, the deterministic adversarial sequence for Parking Permit ensures that the next requested time is later than all the permits the algorithm has purchased so far. Thus, when the Parking Permit request arrives at time t, we pass it to the TCP Acknowledgment algorithm and simulate it into the future to determine at what time t^* it would acknowledge the request, assuming no further requests, and purchase the corresponding permit that covers up to time t^*. Since the next Parking Permit request arrives after the permit expires, the next TCP Acknowledgment request also arrives after t^*. Since the TCP Acknowledgment algorithm is online and the actual request sequence up to time t^* matches the simulation, it also acknowledges at time t^*. Thus, the TCP Acknowledgment algorithm and the Parking Permit algorithm behave identically, yielding the desired lower bound.

Continuous Submodular Delay. The proof of Theorem 1.4 requires a more subtle analysis of Greedy. The second part of the greedy analysis for additive delays breaks down even when the continuous submodular delay cost is a concave function of the total waiting time. Intuitively, this is because the optimal solution can "front-load" its delay cost by acknowledging, e.g., the first few packets a long time after their arrival. This increases the delay cost, so later packets contribute only a small marginal amount to it. To address this, we consider a hypothetical

solution that produces the same acknowledgments as the optimal solution, but each packet's waiting time is capped at its waiting time in Greedy. Since f is monotone, capping the waiting times can only decrease the delay cost. Thus, the cost of the hypothetical solution is at most the cost of the optimal solution, so it suffices to prove a lower bound on the cost of the hypothetical solution. The capping prevents the hypothetical solution from front-loading its delay cost and enables us to argue that if it does not make an acknowledgment during interval I_j, then its delay cost increases by at least 1 during the interval.

2 Preliminaries

Problem Setup. We consider the *Online TCP Acknowledgment* problem. The input is a sequence of packet arrival times $A = (a_1, a_2, \ldots, a_n)$, where a_j is the time at which the j-th packet arrives, and we assume $a_1 \leq a_2 \leq \cdots \leq a_n$. An *online algorithm* must select acknowledgment times $t_1 < t_2 < \cdots < t_k$, such that every packet is acknowledged at or after arrival – that is, for each $1 \leq j \leq n$, there exists some $1 \leq i \leq k$ with $a_j \leq t_i$. Each acknowledgment t_i covers the batch $\sigma_i = \{\, j : t_{i-1} < a_j \leq t_i \,\}$, where we set $t_0 = 0$. These batches form a partition of the packet indices $\{1, \ldots, n\}$. Each packet $j \in \sigma_i$ experiences a *waiting time* (or *delay*) $d_j = t_i - a_j$, and we denote the full delay vector by $d = (d_1, \ldots, d_n)$. The goal is to minimize the total cost, which typically balances two competing objectives: reducing the number of acknowledgments and limiting the total incurred delay. The online nature of the problem is that, as time t increases, the algorithm is aware only of requests whose arrival time is at most t and can only make an acknowledgment at the current time t.

Batch-Aware TCP. The *Batch-Aware TCP* model defines the delay cost in terms of the individual acknowledgment batches. Let $\text{delay}(\sigma_i, t_i)$ denote the delay cost of acknowledging batch σ_i at time t_i. The total delay cost is then computed from the per-batch costs, thereby capturing richer dependencies between the acknowledgment structure and delay.

In this work, we focus on two important classes of batch-aware cost models. In the *sum-monotone delay* model, the total cost is given by $k + \sum_{i=1}^{k} \text{delay}(\sigma_i, t_i)$. Here, the function $\text{delay}(\cdot, \cdot)$ is required to be monotone non-decreasing: if one batch is a subsequence of another and the acknowledgment times are ordered accordingly, the delay cost cannot decrease. This model captures objectives in which the overall delay accumulates across batches and scales with the batch size or delay duration.

In the *max-monotone delay* model, the objective is $k + \max_{1 \leq i \leq k} \text{delay}(\sigma_i, t_i)$, where $\text{delay}(\cdot, \cdot)$ is any monotone non-decreasing function. Unlike the sum-based case, this model focuses on minimizing the worst per-batch delay contribution, which is natural in applications where fairness or tail performance is critical.

Batch-Oblivious TCP. In the *Batch-Oblivious TCP* model, the delay cost is defined globally over the entire delay vector d. Specifically, the delay cost is given by a function $f_n(d_1, \ldots, d_n)$, where $f_n : \mathbb{R}_+^n \to \mathbb{R}_+$ is assumed to be monotone and consistent under zero-padding. Since the number of packets n is not known in advance, we assume the algorithm learns the function f_i only when the ith packet arrives. These functions satisfy the consistency condition

$$f_{i+1}(d_1, \ldots, d_i, 0) = f_i(d_1, \ldots, d_i),$$

ensuring that appending a packet with zero delay does not affect the cost. A natural example of this model is total waiting time, where $f_n(d) = \sum_{j=1}^{n} d_j$. The overall objective in this setting is to minimize $k + f_n(d)$, where k denotes the number of acknowledgments. We refer to this setting as batch-oblivious because the delay cost treats all packet delays globally, regardless of which acknowledgment batch a packet belongs to.

3 Max-Monotone Delay Cost

We begin our investigation with the max-monotone delay model, where the total cost is given by the number of acknowledgments plus the maximum per-batch delay cost. We show that the natural greedy algorithm is 2-competitive.

▶ **Theorem 1.1.** *Greedy is a deterministic 2-competitive algorithm for TCP Acknowledgment with max-monotone delay cost.*

Proof. Recall that we assume the acknowledgment cost is normalized to 1. We analyze the natural greedy algorithm that sends an acknowledgment whenever the total cost increases by 1. Since we are in the max-monotone model, this is equivalent to acknowledging a batch of packets once its delay cost reaches the current batch index. For example, the algorithm sends the first acknowledgment when the currently pending packets accumulate delay cost 1, then the second when the pending batch reaches delay cost 2, and so on. In this way, each acknowledgment increases the total cost by exactly 1 from the delay, in addition to the unit cost of the acknowledgment.

Let t_j denote the time when the algorithm makes its j-th acknowledgment, and define the interval $T_j = (t_{j-1}, t_j]$, with $t_0 = 0$. Moreover, let σ_j be the batch of packets acknowledged at time t_j. By design, we have $\text{delay}(\sigma_j, t_j) = j$, since the algorithm waited until this batch's delay cost reached exactly j. Thus, the total cost of the algorithm is k from acknowledgments and $\max_{1 \le j \le k} \text{delay}(\sigma_j, t_j) = k$ from delay, totaling at most $2k$.

Denote by k^* the number of acknowledgments made by the optimal solution. We now argue that its total cost is at least k, which implies that the greedy algorithm is 2-competitive.

▷ Claim 3.1. If the optimal solution does not send an acknowledgment during interval T_j, then its delay cost is at least j.

Proof. Since no acknowledgment is made during T_j, all packets in σ_j must be acknowledged by the optimal solution in a later batch σ^*, sent at time $t^* \ge t_j$. Because $\text{delay}(\cdot, \cdot)$ is monotone non-decreasing in both batch size and acknowledgment time, it follows that $\text{delay}(\sigma^*, t^*) \ge \text{delay}(\sigma_j, t_j)$, which equals j, proving the claim. ◁

Suppose that the optimal solution makes k^* acknowledgments. If $k^* \ge k$, then we are done. Otherwise, Claim 3.1 implies that the delay cost of the optimal solution is at least $k - k^*$ (as the acknowledgments can be placed in the last k^* T_i intervals) and so the optimal solution has total cost at least k, as desired. ◀

4 Sum-Monotone Delay Cost

In this section, we focus on TCP acknowledgment under the sum-monotone delay-cost model. First, we prove that any deterministic greedy policy is $\Omega(n)$-competitive: there exists an input of size n that enforces the greedy algorithm to pay n-fold more than the offline optimum. We then turn to the offline problem and present a polynomial-time algorithm via dynamic programming, that computes an exact optimum schedule. By embedding this offline oracle into an online framework, we obtain a new algorithm with an $O(\log n)$ competitive ratio: at each decision point, it computes the offline optimum schedule for the set of all packets that have arrived so far and uses that schedule to guide its acknowledgments. Finally, we show that every deterministic algorithm is $\Omega(\log n)$-competitive.

4.1 Lower Bound for Greedy

We begin with a more precise statement of Theorem 1.2. Elsewhere in this paper, when we talk about the Greedy algorithm, we refer to the algorithm that sends an acknowledgment whenever $\text{delay}(\sigma, t) = 1$, where σ is the set of currently unacknowledged packets and t is the current time. Our lower bound holds for a family of greedy algorithms: for $\tau > 0$, algorithm Greedy_τ sends an acknowledgment when $\text{delay}(\sigma, t) = \tau$.

▶ **Theorem 4.1.** *For every $\tau > 0$, the algorithm Greedy_τ is $\Omega(n)$-competitive for TCP Acknowledgment with sum-monotone delay cost.*

Proof. Fix $\tau > 0$. Define the batch delay function $\text{delay}(\sigma, t) = \min\{\sum_{i\in\sigma} d_i, \tau\}$, where $d_i = a_i - t$. Consider the sequence of n packet with arrival times $a_i = i(\tau + \epsilon)$ for a small enough ϵ. Observe that Greedy_τ makes an acknowledgment exactly τ time steps after the arrival of each packet. Thus, Greedy_τ makes n acknowledgments and incurs a delay cost of $n\tau$ for a total cost $n(1 + \tau)$. On the other hand, the solution that waits to make a single acknowledgment at the arrival time of the last packet a_n has a total cost of at most $1 + \tau$. Thus, the competitive ratio of Greedy_τ is at least n. ◀

4.2 Offline Model

Solving the offline version of the TCP acknowledgment under the sum-monotone delay-cost model can be efficiently performed using dynamic programming. The general structure of the solution closely follows the approach used by Dooly et al. [29], though here we consider a general batch cost function $\text{delay}(\sigma, t)$, making our model a natural generalization of their framework.

Let σ be the given sequence of packets, and $\sigma|_i^j$ be its subsequence containing packets i through j. We define the subproblems for our dynamic programming solution as follows. Let $\text{DP}[i]$ denote the minimum total cost of acknowledging packets 1 through i, and set $\text{DP}[0] = 0$. Then, the optimal cost can be computed using the following recurrence relation:

$$\text{DP}[i] = \min_{1\le j\le i} \left\{\text{DP}[j-1] + \text{delay}\left(\sigma|_j^i, a_i\right) + 1\right\}.$$

Intuitively, at each step i, the algorithm evaluates all possible previous acknowledgment positions j and selects the one that minimizes the sum of the optimal cost up to $j - 1$, the delay cost $\text{delay}(\sigma|_j^i, a_i)$, and the acknowledgment cost 1.

The correctness of this approach follows directly from the optimal substructure property: an optimal partitioning of packets into acknowledged subsequences necessarily implies that each of these subsequences itself must be an optimal solution to the corresponding subproblems. The computational complexity of this method is $O(n^2)$, since evaluating each $\text{DP}[i]$ involves examining $O(n)$ previous acknowledgment points, and we have n such subproblems.

4.3 Online Model

Using the offline result from the previous subsection, we now prove the following lemma.

▶ **Lemma 4.2.** *There is a deterministic $O(\log n)$-competitive algorithm for TCP Acknowledgment with sum-monotone delay cost.*

The key idea is to use the cost of the optimal offline solution on the sequence of packets seen so far to guide when our algorithm issues acknowledgments. Suppose the j-th packet arrives and there are no pending packets; we then start a new *budget service s*. Let $\sigma|^j$ denote

the sequence of the first j packets observed so far.[4] We first identify the longest *critical suffix* of $\sigma|^j$, that is, a suffix $\sigma_s \subseteq \sigma|^j$ such that the optimal offline strategy for σ_s consists of issuing a single acknowledgment exactly at the arrival time of its last packet. Observe that at least one such suffix always exists, namely the one containing only packet j. Moreover, notice that for a critical suffix σ_s, the optimal cost of serving it equals $\text{delay}(\sigma_s, a_j) + 1$. To simplify notation, we define $\text{serve}(\sigma_s, a_j) = \text{delay}(\sigma_s, a_j) + 1$. Whenever an acknowledgment is issued at the arrival time of the last packet in the suffix, we omit the second argument and write $\text{serve}(\sigma_s)$.

Based on σ_s, we define a budget b_s for service s (its precise definition will be given later). The service s then aggregates arriving packets and issues an acknowledgment as soon as their total accumulated delay cost reaches b_s; we refer to this moment as the *critical time* of service s. If, during the aggregation phase of a budget service, a new packet arrives, we again compute the longest critical suffix $\sigma' \subseteq \sigma|^j$. If σ' strictly extends σ_s (that is, if it starts earlier than σ_s), we update the budget b_s accordingly to reflect the increased cost suggested by the offline optimum.

When a budget service s finally issues its acknowledgment, the algorithm schedules three consecutive *buffer services*, each with a budget equal to twice the budget of s. Each buffer service aggregates packets and acknowledges them as soon as their delay cost reaches its assigned budget, and this process is repeated three times.[5] During a buffer service, if a packet arrives that induces a critical suffix whose cost is at least twice the cost of σ_s, then the current service is immediately promoted to a new budget service, with its budget defined accordingly.

Assigning exponentially larger budgets to buffer services prevents the algorithm from issuing many low-cost acknowledgments in situations where the offline optimum would issue a single high-cost acknowledgment. By scaling service budgets exponentially in response to increasing costs of critical suffixes, the algorithm ensures that its total cost increases only logarithmically relative to the optimal offline cost, yielding an $O(\log n)$ competitive ratio.

In what follows, we use $\texttt{OPT}(\sigma_s)$ to denote the cost of the optimal offline solution for batch σ_s.

4.3.1 Algorithm Description

We begin by formalizing the notion of *critical batches*, which correspond to the critical suffixes discussed earlier and will play a central role in the design and analysis of our algorithm.

▶ **Definition 4.3** (Critical Batch). *We call batch* σ critical *if it satisfies* $\texttt{OPT}(\sigma) = \text{serve}(\sigma)$.

We are now ready to present Algorithm 1. This algorithm is structured around two event-driven functions: "UponArrival", which is triggered whenever a new packet arrives, and "UponCriticalTime", which is triggered when a precomputed critical time is reached. Together, these functions manage the scheduling of acknowledgments based on the observed input structure.

When a packet arrives, the `UponArrival` appends it to the already-seen packets sequence and finds the largest critical suffix. Then, depending on whether there is a service running and if so – of what type it is – we possibly update the budget of the service (and its type, as in line 11). Notice that line 13 sets the critical time t_c so that acknowledging all currently

[4] Here we slightly overload notation: earlier, $\sigma|^j$ denoted the j-th batch served by the algorithm.
[5] The purpose of these buffer services is to separate consecutive budget services and simplify the analysis.

Algorithm 1 Sum-Monotone TCP Algorithm.

Initialization: sequence of already seen packets $\sigma = []$, critical time $t_c = 0$, critical batch $\sigma_c = []$ and budget b

Function UponArrival(r):

1: put r at the end of σ
2: let σ' be the largest critical suffix
3: **if** there is no service running, i.e., σ_c is empty **then** ▷ start a buffer service
4: $\sigma_c = \sigma'$
5: **else if** there is a budget service running **then**
6: **if** $\sigma_c \subseteq \sigma'$ **then** ▷ check if the current critical suffix is a larger
7: $\sigma_c = \sigma'$ ▷ if so, update the critical batch
8: **else if** there is a buffer service running **then**
9: **if** $\text{serve}(\sigma') \geq 2 \cdot \text{serve}(\sigma_c)$ **then** ▷ check whether the critical suffix is large enough
10: $\sigma_c = \sigma'$ ▷ if so, update the critical batch
11: switch the current service to a budget service
12: $b = 2 \cdot \text{serve}(\sigma_c)$ ▷ update the critical batch
13: let t_c be such that $\text{serve}(\text{pending packets}, t_c) = b$ ▷ update the critical time

Function UponCriticalTime(): ▷ called when t_c is reached

1: make an acknowledgment
2: **if** the budget service ended **then**
3: start a new buffer service and double the budget $b \leftarrow 2b$
4: **else if** the first or second buffer service ended **then**
5: start a consecutive buffer service
6: **else if** the third buffer service ended **then**
7: $\sigma_c = []$, $b = 0$ ▷ reset the variables

pending packets at time t_c will incur a delay cost exactly equal to the budget b. If this cost never reaches b, we assume $t_c = \infty$, meaning the current service will conclude only at the end of the input sequence.

The `UponCriticalTime` function is called when t_c is reached. It performs an acknowledgment and issues a new service, or resets the state if all buffer services have completed.

4.3.2 Critical Batches and Relations Between Them

We begin by introducing some notation. Suppose that Algorithm 1 issues k acknowledgments for budget services over a sequence of n requests. We denote the corresponding critical batches by σ_c^i for $i \in \{1, 2, \ldots, k\}$. Each critical batch may be followed by up to three buffer services. We denote by $\sigma_{b,l}^i$, for $l \in \{1, 2, 3\}$, the sequences of packets acknowledged by these services. If a particular buffer service is not started, we define the corresponding batch to be the empty sequence.

▶ **Definition 4.4** (Critical Batch Intersection). *We say that the critical batch σ_c^j* intersects *σ_c^i, $i < j$, if*

(1) *$j = i + 1$ and the budget service corresponding to σ_c^j was started in line 11 of the `UponArrival` function in Algorithm 1, or*

(2) $\sigma^i_{b,2}$ *and* $\sigma^i_{b,3}$ *are both not empty, while* σ^j_c *and* $\sigma^i_{b,2}$ *share at least one packet, that is,* $\sigma^j_c \cap \sigma^i_{b,2} \neq \emptyset$.

When using set-theoretic notation, we implicitly identify a sequence with the set of packets it contains.

Understanding how critical batches intersect is key to analyzing the proposed algorithm. To make our reasoning simpler, we introduce a few new terms to describe this relation.

▶ **Definition 4.5** (Parent Batch). *Given a critical batch* σ^i_c*, we define the set* $\texttt{children}(\sigma^i_c)$ *to consist of all earlier critical batches that* σ^i_c *intersects and for which* σ^i_c *is the first subsequent critical batch to do so. For each batch* $\sigma^j_c \in \texttt{children}(\sigma^i_c)$*, we refer to* σ^i_c *as its* parent batch*. We use the terms* ancestor *and* descendant *to denote the transitive closure of this relation.*

▶ **Definition 4.6** (Final Critical Batch). *We call a batch with no parent a* final critical batch.

In what follows, we also use the following notation for the pivotal packet arrival times. We use t^i_s and t^i_c to denote the first and the last packet's arrival times in σ^i_c. Moreover, if σ^i_c is a parent batch, we let σ^i_e be its descendant that was acknowledged the earliest, and let t^i_e be the arrival of the first packet in σ^i_e.

▶ **Definition 4.7** (Coverage Interval). *Given a critical batch* σ^i_c*, we define its* coverage interval *as follows. If* σ^i_c *has no children, then its coverage interval is* $[t^i_s, t^i_b]$*; otherwise, it is* $[t^i_e, t^i_b]$*. We denote this interval by* I^i_c*. When convenient, we also use* I^i_c *to denote the set of packets arriving within this time interval.*

4.3.3 Analysis

Let us start by showing how final critical batches relate to the optimal cost of serving packets.

▶ **Lemma 4.8.** *Let* $\sigma^1_c, \ldots, \sigma^\ell_c$ *be a sequence of final critical batches obtained during a run of Algorithm 1 on a packet sequence* $\hat{\sigma}$*. We assume that they are listed in the order in which the corresponding budget services were issued, and that the list contains all such batches. Then, it holds that*

$$\texttt{OPT}\left(\hat{\sigma}\right) \geq \sum_{i=1}^{\ell} \texttt{OPT}\left(I^i_c\right).$$

Proof. Let $\hat{\sigma}_1, \ldots, \hat{\sigma}_p$ denote the partition of $\hat{\sigma}$ induced by the optimal solution, where each $\hat{\sigma}_i$ corresponds to a single acknowledgment. By definition, each $\hat{\sigma}_i$ forms a critical batch and therefore must be recognized by Algorithm 1.

Suppose that for some $j \in \{1, \ldots, p\}$, the batch $\hat{\sigma}_j$ overlaps with at least two coverage intervals corresponding to the final critical batches identified by the algorithm. Let σ^k_c, for some $k \in \{1, \ldots, \ell\}$, be the latest such critical batch, and let $\hat{j}$ denote the last packet in $\hat{\sigma}_j$. Finally, let $\sigma^{k,j}_c$ be the largest critical batch among σ^k_c and its descendants such that $\hat{j}$ belongs to one of the services associated with this batch.

If $\hat{j}$ is served by the budget service associated with $\sigma^{k,j}_c$, then at the moment $\hat{j}$ arrives, the `UponArrival` function would recognize $\hat{\sigma}_j$ in either line 4 or 7 and act upon it. Otherwise, if $\hat{j}$ is served by a buffer service, it would trigger the start of a new budget service in line 11. Indeed, since $\hat{\sigma}_j$ overlaps with all packets acknowledged by the budget service corresponding to $\sigma^{k,j}_c$, its service cost exceeds $2 \cdot \text{serve}(\sigma^{k,j}_c)$, as stated in line 12, and thus the condition in line 9 is satisfied.

Therefore, no batch served by the optimal solution can overlap with more than one coverage interval. Consequently, for each coverage interval I_c^i, we consider the set of all batches $\hat{\sigma}_j$ that overlap with it. This set covers at least as many packets as I_c^i, and hence its total cost is at least $\texttt{OPT}(I_c^i)$. Since each batch $\hat{\sigma}_j$ is counted at most once, the desired inequality follows. ◀

From there, we obtain two straightforward properties.

▶ **Corollary 4.9.** *The total cost of* OPT *on a given sequence of packets σ is at least equal to the sum of the optimal costs of serving coverage intervals of all the final critical batches recognized during the run of Algorithm 1 on σ.*

▶ **Corollary 4.10.** *For any critical batch with children, the optimal cost of serving its coverage interval is at least the sum of the children's critical interval optimal serving costs.*

Now, observe that also the following holds for any critical batch with children.

▶ **Lemma 4.11.** *Let σ_c^i be a critical batch such that* $\texttt{children}(\sigma_c^i) \neq \emptyset$*. Then, for any child batch σ_c^j, we have* $\text{serve}(\sigma_c^i) \geq 2 \cdot \text{serve}(\sigma_c^j)$.

Proof. Since σ_c^i is a parent of σ_c^j, Definition 4.5 implies that σ_c^i intersects σ_c^j. Therefore, one of the two conditions in Definition 4.4 – namely, 1 or 2 – must hold.

We first consider condition 1. In this case, we have $j = i + 1$, and the budget service corresponding to σ_c^j is initiated in line 11 of the `UponArrival` function by some critical suffix σ'. Consequently, the condition in line 9 holds, which implies that $\text{serve}(\sigma') \geq 2 \cdot \text{serve}(\sigma_c^i)$. Since $\sigma' \subseteq \sigma_c^j$ (as σ' may have been extended in line 6 to become σ_c^i), the claimed inequality follows.

We now turn to condition 2. In this case, both $\sigma_{b,2}^i$ and $\sigma_{b,3}^i$ are non-empty, and σ_c^j shares at least one packet with $\sigma_{b,2}^i$, that is, $\sigma_c^j \cap \sigma_{b,2}^i \neq \emptyset$. Since σ_c^j overlaps with $\sigma_{b,2}^i$, it must also cover $\sigma_{b,3}^i$, which immediately follows $\sigma_{b,2}^i$. As $\sigma_{b,3}^i \neq \emptyset$, its service cost satisfies $\text{serve}(\sigma_{b,3}^i) = 2 \cdot 2 \cdot \text{serve}(\sigma_c^i)$, by lines 3 of the `UponCriticalTime` function and 12 of the `UponArrival` function. Hence, $\text{serve}(\sigma_c^j) \geq \text{serve}(\sigma_{b,3}^i) = 4 \cdot \text{serve}(\sigma_c^i)$, and the desired inequality also holds in this case. ◀

▶ **Lemma 4.12.** *The budget service associated with the critical batch σ_c^i incurs a cost of $2 \cdot \text{serve}(\sigma_c^i)$, while each buffer service associated with σ_c^i incurs a cost of at most $4 \cdot \text{serve}(\sigma_c^i)$.*

Proof. By the definition of the algorithm. ◀

▶ **Lemma 4.13.** *Every final critical batch σ_c^i has at most $\log(\texttt{OPT}(I_c^i)) \leq \log n$ levels of descendant sets.*

Proof. By Lemma 4.11, each child batch has a cost at least twice that of the parent. Since the minimum cost is 1 (the acknowledgment cost), the first property follows. To observe the upper bound on this value, notice that one of the possible solutions is to acknowledge each packet individually, and thus $\texttt{OPT}(I_c^i) \leq n$. ◀

Combining the above lemmas gives us Lemma 4.2. Indeed, by Lemma 4.13 and a recursive application of Corollary 4.10, we have that the sum of optimal costs for all the critical batches processed by our algorithm (i.e., those for which a service was issued) is at most equal to $\log n \cdot \sum_{i=1}^{\ell} \texttt{OPT}(I_c^i)$. Thus, by Lemma 4.12, the algorithm paid at most $14 \log n \cdot \sum_{i=1}^{\ell} \texttt{OPT}(I_c^i)$ to serve all the packets. The lower bound on optimal cost from Corollary 4.9 completes the proof.

4.4 Lower Bounds for Arbitrary Algorithms

We show that the deterministic $O(\log n)$ competitive ratio of our algorithm (Algorithm 1) is tight. We start by defining the Parking Permit Problem [44]. It is known that for every deterministic algorithm $\mathcal{B}$ for Parking Permit, there is an adversarial request sequence A on which $\mathcal{B}$ is $\Omega(\log n)$-competitive (Lemma 4.14 [46]). Then, we show how to convert any deterministic algorithm $\mathcal{A}$ for TCP Acknowledgment into a deterministic algorithm $\mathcal{B}_{\mathcal{A}}$ for Parking Permit. Finally, we show that $\mathcal{A}$ is $\Omega(\log n)$-competitive on the adversarial sequence for $\mathcal{B}_{\mathcal{A}}$ (Lemma 4.17).

Parking Permit Problem [44]. An instance of the Parking Permit Problem. For each timestep $t \geq 1$, a request may arrive, and if so, needs to be covered by a permit. There are several different types of permits that can be purchased at any time throughout the time horizon. A permit of type $k \geq 0$ has cost C_k and duration D_k: when bought at time t, it covers the time interval $[t, t + D_k]$. The goal is to minimize the cost of the purchased permits.

Hard Parking Permit instances. A *hard Parking Permit instance* is one where $C_k = 2^k$ and $D_k = 4^k$ for each $k \geq 0$.

▶ **Lemma 4.14** ([46]). *Let $\mathcal{B}$ be a deterministic online algorithm for Parking Permit. Consider a hard Parking Permit instance where the sequence of request arrival times A is generated by the following adversary: for each $1 \leq j \leq n$ (in ascending order), request j arrives at the earliest timestep that is not yet covered by the permits bought by $\mathcal{B}$.*

When run on A, algorithm $\mathcal{B}$

- *buys a separate permit $[a_j, t_j]$ for each request j; moreover, $a_{j+1} = t_j + 1$ for each $j < n$;*
- *has competitive ratio $\Omega(\log n)$.*[6]

Hard TCP Acknowledgment instances. We now define the corresponding TCP Acknowledgment instances. Define the piecewise-linear concave function $f(x) = \min_{k \geq 0} C_k + x \cdot (C_k/D_k)$ where C_k and D_k are defined as above. A *hard TCP Acknowledgment instance* is one in which the total cost of a batch σ acknowledged at time t – i.e., sum of the acknowledgment cost[7] and the delay cost of the batch $\text{delay}(\sigma, t)$ – is given by $f(t - a(\sigma))$ where $a(\sigma)$ is the earliest arrival time among the packets in σ. That is, $\text{delay}(\sigma, t) = f(t - a(\sigma)) - 1$.

Converting TCP Acknowledgment algorithm to Parking Permit. Observe that a hard Parking Permit instance with arrival sequence A has a corresponding hard TCP Acknowledgment instance with the same arrival sequence. Our goal now is to show how to map decisions of a TCP Acknowledgment algorithm on the corresponding instance into decisions for the Parking Permit instance. We begin by presenting the two necessary ingredients.

First, we need to decide which permit to buy when a request arrives even though the TCP Acknowledgment algorithm is not required to commit on arrival of a request, when the request will be acknowledged. For a sequence A, let A_j denote the subsequence containing the first j elements. The following lemma, together with Proposition 4.16, lets us decide which permit to buy on arrival of a request.

[6] We remark that the Parking Permit Problem is typically defined as having a time horizon $[1, N]$ (on which n requests arrive) and Lemma 21 of [46] gives a lower bound of $\Omega(\log N)$. Since $N \geq n$, we get a lower bound of $\Omega(\log n)$.

[7] Recall that the acknowledgment cost is 1.

▶ **Lemma 4.15.** *Let $\mathcal{A}$ be a deterministic online algorithm for TCP Acknowledgment. Then, there exists a function* next *such that for every sequence of packet arrival times $A = (a_1, \ldots, a_n)$ and for every j, if $a_{j+1} > \text{next}(A_j)$, then packet j is acknowledged at time $\text{next}(A)$. The function* next *is called the* threshold function *of $\mathcal{A}$.*

Proof. Fix a sequence A and $1 \leq j \leq n$. Define $\text{next}(A_j) \geq a_j$ to be the time that $\mathcal{A}$ acknowledges the last packet of A_j when $\mathcal{A}$ is run on A_j. If $a_{j+1} > \text{next}(A_j)$, then up till time $\text{next}(A_j)$, the executions of $\mathcal{A}$ on A_j and of $\mathcal{A}$ on A see the same information. Since $\mathcal{A}$ is an online algorithm, the execution of $\mathcal{A}$ on A will also acknowledge at time $\text{next}(A_j)$. ◀

Second, we need to be able to map batches to permits. In general, the time between when a batch is acknowledged and the earliest arrival time of the batch can be arbitrary, while permits can only have lengths $D_k = 4^k$. The following lets us approximately map batches to permits.

▶ **Proposition 4.16.** *For every $x \geq 0$, there exists $k^* \geq 0$ such that $D_{k^*} \geq x$ and $C_{k^*} \leq 2f(x)$.*

Proof. Suppose $f(x) = C_k + x(C_k/D_k)$. If $x < D_k$, then $f(x) \geq C_k$ so choosing $k^* = k$ works. Next, we consider the case when $x > D_k$. Let $k^* > k$ be such that $D_{k^*-1} \leq x \leq D_{k^*}$. Observe that $C_k/D_k = (1/2)^k \leq (1/2)^{k^*-1} = C_{k^*-1}/D_{k^*-1}$. Together with the fact that $x \geq D_{k^*-1}$, we have

$$f(x) \geq x(C_k/D_k) \geq D_{k^*-1}(C_{k^*-1}/D_{k^*-1}) = C_{k^*-1} = C_{k^*}/2,$$

as desired. ◀

We now have the ingredients to convert any deterministic algorithm $\mathcal{A}$ for hard TCP Acknowledgment instances to a deterministic algorithm $\mathcal{B}_\mathcal{A}$ for hard Parking Permit instances. Let $\text{next}(\cdot)$ be the threshold function of $\mathcal{A}$. The algorithm $\mathcal{B}_\mathcal{A}$ works as follows. For each request arrival a_j that is not covered by permits bought so far:

(1) let A_j be the sequence of arrival times so far, including a_j;
(2) let k^* be such that $D_{k^*} \geq x$ and $f(D_{k^*}) \leq 2f(\text{next}(A_j) - a_j)$ as guaranteed by Proposition 4.16;
(3) buy permit $[a_j, a_j + D_{k^*}]$.

▶ **Lemma 4.17.** *Let $\mathcal{A}$ be a deterministic algorithm for hard TCP Acknowledgment instances and A^* be the adversarial sequence for $\mathcal{B}_\mathcal{A}$ given by Lemma 4.14. Then, $\mathcal{A}$ is $\Omega(\log n)$-competitive on A^*.*

Proof. Let $\texttt{OPT}_{\text{PP}}(A^*)$ and $\texttt{OPT}_{\text{TCP}}(A^*)$ be the cost of the optimal Parking Permit and TCP Acknowledgment solutions to A^*, respectively. Since Lemma 4.14 already shows that $\mathcal{B}_\mathcal{A}$ is $\Omega(\log n)$-competitive on A^*, it suffices to show the cost of $\mathcal{A}$ on A^* is at least $\Omega(1)$ times the cost of $\mathcal{B}_\mathcal{A}$ on A^*, and that $\texttt{OPT}_{\text{TCP}}(A^*) = O(1)\texttt{OPT}_{\text{PP}}(A^*)$.

By Lemma 4.14 and by definition of $\mathcal{B}_\mathcal{A}$, for every $j < n$, we have that $a_{j+1} > a_j + D_{k^*} \geq \text{next}(A_j)$. Thus, for each request j, Lemma 4.15 implies that $\mathcal{A}$ acknowledges j by itself at time $\text{next}(A_j)$ and the total cost of the batch is $f(\text{next}(A_j) - a_j))$; on the other hand, $\mathcal{B}_\mathcal{A}$ buys permit $[a_j, a_j + D_{k^*}]$ of cost $C_{k^*} \leq 2f(\text{next}(A_j) - a_j)$. Thus, the cost of $\mathcal{A}$ on A^* is at least $1/2$ the cost of $\mathcal{B}_\mathcal{A}$ on A^*.

Next, we show that $\texttt{OPT}_{\text{TCP}}(A^*) = O(1)\texttt{OPT}_{\text{PP}}(A^*)$. Let $P_i = [s_i, t_i]$ be the i-th permit bought by the optimal solution for the hard instance of Parking Permit with arrival sequence A. We construct a solution to the hard TCP Acknowledgment instance as follows: while there is an unacknowledged packet, let a_j be the earliest unacknowledged packet and send an acknowledgment at time t where t is the furthest end time among all permits covering a_j (at least one such permit exists since the permits have to cover A).

We now argue that we can charge the total cost of each batch to a unique permit. For each batch σ_i acknowledged at time t_i, let $P_{(i)}$ be the permit that covers $a(\sigma_i)$ and has end time t_i. Since $P_{(i)}$ is the permit with the furthest end time, we have that $P_{(i')} \neq P_{(i)}$ for $i' \neq i$. Let $C_{(i)}$ and $D_{(i)}$ be the cost and duration of $P_{(i)}$. Since $a(\sigma_i) \in P_{(i)}$, we get that the total cost of σ_i is $f(t_i - a(\sigma_i)) \leq f(D_{(i)}) = 2C_{(i)}$. Thus the cost of each batch σ_i is charged to at most twice the cost of permit $P_{(i)}$, and each permit is charged by at most one batch. Thus, we have shown that $\texttt{OPT}_{\text{TCP}}(A^*) = O(1)\texttt{OPT}_{\text{PP}}(A^*)$, completing the proof of the lemma. ◀

Thus, we conclude that every deterministic algorithm for TCP Acknowledgment with sum-monotone delay costs is $\Omega(\log n)$-competitive, as desired. This also completes the proof of Theorem 1.3.

5 Batch-Oblivious Delay Costs

We now turn to the batch-oblivious model, where the delay cost is defined globally over the vector of packet delays, without reference to how packets are grouped in acknowledgments. Formally, the objective is to minimize $k + f(d)$, where k is the number of acknowledgments, $f : \mathbb{R}_+^n \to \mathbb{R}_+$ is a monotone delay function, and $d = (d_1, \ldots, d_n)$ is the vector of waiting times, where d_i denotes how long packet i waits before being acknowledged. The function f is revealed incrementally as packets arrive. While this framework captures natural objectives such as total waiting time, it also includes more general functions, such as concave or submodular delay costs. As discussed earlier, greedy algorithms that succeed for additive delay fail in these settings, due to the ability of the optimal solution to concentrate delay on a few early packets and reduce marginal costs for the rest. A formal lower bound for concave delay functions is provided in Section 5.1. Here, we focus on broader classes of delay functions, including submodular functions and norms.

5.1 Concave Delay Cost

▶ **Theorem 1.6.** *Every deterministic algorithm is $\Omega(\sqrt{n})$-competitive for TCP Acknowledgment with batch-oblivious concave delay cost.*

Proof. Let $\ell \leq n/2$ and $\epsilon \in [0, 1]$ be parameters that will be set later. Define $x, y \in \mathbb{R}^n$ such that: each of the first ℓ coordinates of x is ϵ and each of the remaining coordinates is 1; each of the first ℓ coordinates of y is n/ℓ and each of the remaining is ϵ. Given a waiting time vector d, define $d_{\leq \ell} = \sum_{i \leq \ell} d_i$ and $d_{>\ell} = \sum_{i>\ell} d_i$. We now define the delay function

$$f(d) = \min\{d \cdot x, d \cdot y\} = \min\left\{\epsilon d_{\leq \ell} + d_{>\ell}, \frac{n}{\ell} d_{\leq \ell} + \epsilon d_{>\ell}\right\}.$$

Note that the function f is concave as it is the point-wise minimum of linear functions.

The adversary proceeds as follows. For the first ℓ packets, it releases packet i at time i. Let A_ℓ be the number of packets made by the algorithm just before time $\ell + 1$. If $A_\ell \geq \ell/2$ (case 1), it releases the remaining $n - \ell$ packets together at time $\ell + 1$; otherwise (case 2), it continues releasing each remaining packet i at time i.

We now show that in both cases, the adversary can construct a significantly cheaper solution. Consider case 1: the algorithm incurs a cost of at least $\ell/2$. Meanwhile, the adversary can use a single acknowledgment at time $\ell + 1$. Let d^* denote the resulting waiting time vector, where $d_i^* = \ell + 1 - i$ for $i \leq \ell$, and $d_i^* = 0$ for $i > \ell$. The delay cost of this solution is at most $f(d^*) \leq \epsilon \sum_{i=1}^{\ell} i = O(\ell^2 \epsilon)$, and the total cost is $1 + O(\ell^2 \epsilon)$. Therefore, the competitive ratio of the algorithm is $\Omega\big(\min\{\ell, 1/(\ell\epsilon)\}\big)$ in this case.

Next, we consider case 2. Let k denote the total number of acknowledgments made by the algorithm. Without loss of generality, we may assume that each acknowledgment occurs at the arrival time of some packet, as this can only reduce the total cost. Recall that in this case, packets arrive at unit times $1, 2, \ldots, n$. Thus, for any time interval starting at a packet arrival, the number of acknowledgments made during the interval plus the total waiting time incurred by the packets arriving within it must be at least the length of the interval. This is because the acknowledgment cost is 1, so each acknowledgment can "pay for" at most one unit of waiting time.

Now, consider the first ℓ packets. By assumption, the algorithm makes $A_\ell < \ell/2$ acknowledgments during this interval. From our previous observation, it follows that the total delay incurred by these packets is at least $\ell - A_\ell > \ell/2$. Thus, we obtain that $d \cdot y \geq (n/\ell) \cdot (\ell/2) = \Omega(n)$. On the other hand, consider only the remaining $n - \ell$ packets. Here, the number of acknowledgments and the overall waiting time sum up to $(k - A_\ell) + d_{>\ell}$. As noted above, this sum must be at least the length of the interval $(\ell, n]$, which is greater than $n/2$, since $\ell < n/2$. Hence, we have $k + d \cdot x \geq (k - A_\ell) + d_{>\ell} = \Omega(n)$. Thus, the total cost of the algorithm satisfies $k + \min\{d \cdot x, d \cdot y\} \geq \Omega(n)$.

On the other hand, consider the solution that acknowledges each of the first ℓ packets on arrival and makes one acknowledgment at the end at time n. Let d^* be its waiting time vector. Then, it holds that $f(d^*) \leq d^* \cdot y = \epsilon d_{>\ell} \leq O(n^2\epsilon)$ and so the solution has total cost $\ell + O(n^2\epsilon)$. Thus, the competitive ratio is $\Omega(\min\{n/\ell, 1/(n\epsilon)\})$ in this case.

Setting $\ell = \lceil \sqrt{n} \rceil$ and $\epsilon = 1/n^2$ gives the theorem. ◀

5.2 Submodular Delay Cost and Norms

We begin by defining continuous submodularity.

▶ **Definition 5.1** (Continuous submodular functions). *A function $f : \mathbb{R}^n_+ \to \mathbb{R}_+$ is* continuous submodular *if for every $x, y \in \mathbb{R}^n$, it satisfies $f(x \vee y) + f(x \wedge y) \leq f(x) + f(y)$, where $\vee$ denotes coordinate-wise maximum and $\wedge$ denotes coordinate-wise minimum.*

We remark that, unlike the case of discrete submodular functions, continuous submodularity is only equivalent to a weaker notion of diminishing marginals: for every $x, y \in \mathbb{R}^n_+$ such that $x \leq y$, $i \in [n]$ such that $x_i = y_i$ and $a \geq 0$, we have $f(x + ae_i) - f(x) \geq f(y + ae_i) - f(y)$, where e_i is the i-th standard basis vector. We refer the reader to [47, Remark 7 and Appendix A.1] for further discussion on this topic.

In fact, our analysis only uses an even weaker condition, where the diminishing-marginals property is required only when the coordinate being increased is zero in both vectors. We say that f satisfies *zero-coordinate diminishing marginals* if for every $x, y \in \mathbb{R}^n_+$ such that $x \leq y$, every $i \in [n]$ such that $x_i = y_i = 0$, and every $a \geq 0$, we have $f(x + ae_i) - f(x) \geq f(y + ae_i) - f(y)$. Clearly, continuous submodularity implies this condition.

For a vector $\Delta \in \mathbb{R}^n_+$, let $\mathrm{supp}(\Delta) = \{i \in [n] : \Delta_i > 0\}$. We will use the following simple observation.

▶ **Observation 5.2.** *Suppose that f satisfies zero-coordinate diminishing marginals. Let $u, v, \Delta \in \mathbb{R}^n_+$ be such that $u \leq v$ and $u_i = v_i = 0$ for every $i \in \mathrm{supp}(\Delta)$. Then*

$$f(u + \Delta) - f(u) \geq f(v + \Delta) - f(v).$$

Indeed, one can add the coordinates in $\mathrm{supp}(\Delta)$ one at a time. Before each coordinate is added, it is still zero in both the smaller and the larger vector, so zero-coordinate diminishing marginals apply. Summing these inequalities over all coordinates in $\mathrm{supp}(\Delta)$ gives the desired inequality.

▶ **Theorem 1.4.** *Greedy is a deterministic 2-competitive algorithm for TCP Acknowledgment with batch-oblivious continuous submodular delay cost.*

Proof. Recall that Greedy makes an acknowledgment whenever its delay cost increases by 1. We now analyze the competitive ratio of the algorithm. First, observe that by construction, the cost of the algorithm is at most twice the number of acknowledgments. Let k be the number of acknowledgments made by the algorithm. Fix an optimal solution. In the rest of the proof, we argue that its cost is at least k.

Let t_j be the time when the algorithm made its j-th acknowledgment, and define $T_j = (t_{j-1}, t_j]$ with $t_0 = 0$. For each packet i, let $d_i(j)$ and $d_i^*(j)$ be the amount of waiting time that packet i incurred up till t_j in the algorithm's solution and the optimal solution, respectively. If packet i arrived after t_j, we define $d_i(j)$ and $d_i^*(j)$ to be both 0.

At a high level, we adopt the proof framework of [29] for the standard TCP Acknowledgment setting and generalize it to our case: within each interval T_j, the optimal solution either issues an acknowledgment and pays a cost of 1 or its delay cost increases by at least 1. However, if the optimal solution allows earlier packets to wait a long time, it can effectively "front-load" its delay cost, which causes the marginal cost in later intervals to be too small. To address this, we introduce a capped version of the optimal delay vector, denoted $\bar{d}$, where each entry is defined as $\bar{d}_i(j) = \min\{d_i^*(j), d_i(j)\}$. In what follows, we use $d(j)$ and $\bar{d}(j)$ to denote the vectors $(d_1(j), \ldots, d_n(j))$ and $(\bar{d}_1(j), \ldots, \bar{d}_n(j))$, respectively. By monotonicity, the optimal solution's delay cost $f(d^*) \geq f(\bar{d})$ and so it suffices to show that if the optimal solution makes no acknowledgment in T_j, then the increase in the delay cost of the capped optimal delay vector in T_j is $f(\bar{d}(j)) - f(\bar{d}(j-1)) \geq 1$.

▷ Claim 5.3. For each interval T_j, if the optimal solution did not make an acknowledgment in T_j, then

$$f(\bar{d}(j)) - f(\bar{d}(j-1)) \geq f(d(j)) - f(d(j-1)) = 1.$$

Proof. The equality follows from the fact that $f(d(j)) - f(d(j-1))$ is the increase in the algorithm's delay cost during interval T_j, and the algorithm makes an acknowledgment exactly when its delay cost increases by 1 since the last acknowledgment.

It remains to prove the inequality. Let $\Delta^j = d(j) - d(j-1)$ be the increase in the algorithm's delay vector during interval T_j. Since the algorithm acknowledged all pending packets at time t_{j-1}, the only packets whose delay can increase during T_j are the packets that arrive during T_j. Hence, for every $i \in \mathrm{supp}(\Delta^j)$, we have $d_i(j-1) = 0$. Moreover, by the definition of the capped vector, $\bar{d}_i(j-1) = 0$. We also have $\bar{d}(j-1) \leq d(j-1)$. Therefore, applying Observation 5.2 with $u = \bar{d}(j-1)$, $v = d(j-1)$, and $\Delta = \Delta^j$, we get

$$f(\bar{d}(j-1) + \Delta^j) - f(\bar{d}(j-1)) \geq f(d(j-1) + \Delta^j) - f(d(j-1)).$$

Since $d(j-1) + \Delta^j = d(j)$, this gives

$$f(\bar{d}(j-1) + \Delta^j) - f(\bar{d}(j-1)) \geq f(d(j)) - f(d(j-1)). \tag{2}$$

It remains to compare $\bar{d}(j-1) + \Delta^j$ with $\bar{d}(j)$. We claim that $\bar{d}(j-1) + \Delta^j \leq \bar{d}(j)$ coordinatewise. Fix a packet i. If i arrived during T_j, then, since the optimal solution did not make an acknowledgment in T_j, the packet has the same waiting time up till t_j in both solutions. Thus, $d_i^*(j) = d_i(j)$, and hence $\bar{d}_i(j) = d_i(j) = \Delta_i^j$. Also, $\bar{d}_i(j-1) = 0$, so the desired inequality holds with equality.

If packet i arrived no later than t_{j-1}, then the algorithm does not incur additional waiting time for packet i during T_j, so $\Delta_i^j = 0$. Since waiting times in the optimal solution do not decrease while no acknowledgment is made, and since $d_i(j) = d_i(j-1)$, we have

$\bar{d}_i(j-1) \le \bar{d}_i(j)$. Thus, the desired inequality again holds. Finally, if packet i arrives after t_j, all relevant quantities are 0, and the inequality is trivial. Therefore, $\bar{d}(j-1) + \Delta^j \le \bar{d}(j)$, as desired.

Applying monotonicity of f to the above inequality, we have $f(\bar{d}(j)) \ge f(\bar{d}(j-1) + \Delta^j)$. Plugging this into Equation (2) gives

$$f(\bar{d}(j)) - f(\bar{d}(j-1)) \ge f(d(j)) - f(d(j-1)).$$

Together with the greedy rule $f(d(j)) - f(d(j-1)) = 1$, this proves the claim. ◁

Suppose the optimal solution makes k^* acknowledgments. If $k^* \ge k$, then we are done. Otherwise, there exist at least $k - k^*$ intervals in which the optimal solution did not make an acknowledgment. By monotonicity of f and Claim 5.3, the optimal solution's delay cost is

$$f(d^*) \ge f(\bar{d}) = \sum_{j=1}^{k} f(\bar{d}(j)) - f(\bar{d}(j-1)) \ge k - k^*.$$

Thus, the optimal solution's total cost is at least k. ◀

Patton, Russo, and Singla [47] observed that ℓ_p norms, Top-k norms and ordered norms are submodular. They also proved a result on the approximability of symmetric norms by submodular norms. Using these results give us the following corollary.

▶ **Corollary 1.5.** *There is a deterministic 2-competitive algorithm for TCP Acknowledgment, where the delay cost is: an ℓ_p norm, a Top-ℓ norm, or an ordered norm. When the delay cost is a symmetric norm $\|\cdot\|$, then there is a deterministic $O(\log \rho)$-competitive algorithm where $\rho = \|(1,1,\ldots,1)\| \,/\, \|(1,0,\ldots,0)\| \le n$. In particular, there is a deterministic $O(\log n)$-competitive algorithm when the delay cost is a monotone symmetric norm.*

▶ Remark 5.4. The zero-coordinate diminishing-marginals condition has a natural modeling interpretation in practical domains beyond TCP Acknowledgment. It says that the marginal cost of making a new coordinate positive is smaller when more other coordinates are already present. This is appropriate for incident-based or SLA-type[8] delay costs, where the first few delayed requests may create most of the operational or reputational cost, while additional newly delayed requests are partly absorbed into the same incident. For example, functions of the form $f(d) = g(\mathrm{supp}(d))$, where g is a monotone submodular function, satisfy zero-coordinate diminishing marginals: adding a new positive coordinate corresponds exactly to adding a new element to a larger set, whose marginal contribution is smaller by the submodularity of g. Such functions model global costs depending on the set of requests that experience any positive delay, rather than on the precise magnitude of each delay.

6 Conclusion

In this work, we revisit Online TCP Acknowledgment under broad and natural generalizations of delay costs. We show that the classical guarantees of Greedy extend well beyond additive delays: it remains 2-competitive for *max-monotone* objectives and for a rich class of *batch-oblivious* continuous submodular delay costs. At the same time, we identify a sharp boundary where this paradigm breaks, by proving that Greedy can be $\Omega(n)$-competitive for *sum-monotone* delays. For this harder regime, we develop a new phase-based online algorithm

[8] SLA = Service Level Agreement.

that achieves a tight $\Theta(\log n)$ competitive ratio. Together, our results give a more complete picture of TCP under non-additive delay objectives and lay conceptual groundwork for online problems with delay more broadly.

There are several intriguing open problems. First, what is the competitiveness of other problems such as Joint Replenishment, Multi-Level Aggregation, and Set Cover with Delay under the more general delay cost models considered in this work? Second, while we have a tight characterization of the deterministic competitive ratio for sum-monotone delays, it is unclear whether randomization can help. For the Parking Permit problem, Meyerson [44] showed that the randomized competitive ratio is $\Theta(\log \log n)$. A reduction from Parking Permit will imply the same lower bound. Finally, are there other interesting delay objectives? For example, fairness-oriented delay objectives, where the goal is to balance efficiency with equitable treatment of packets, a setting that remains largely unexplored in the TCP acknowledgment framework.

References

1 Susanne Albers. Online algorithms: a survey. *Mathematical Programming*, 97:3–26, 2003. doi:10.1007/S10107-003-0436-0.

2 Susanne Albers and Helge Bals. Dynamic TCP acknowledgement: penalizing long delays. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 12-14, 2003, Baltimore, Maryland, USA*, pages 47–55. ACM/SIAM, 2003. URL: http://dl.acm.org/citation.cfm?id=644108.644116.

3 Susanne Albers and Helge Bals. Dynamic TCP acknowledgment: Penalizing long delays. *SIAM J. Discret. Math.*, 19(4):938–951, 2005. doi:10.1137/S0895480104441656.

4 Mark Allman. On the generation and use of tcp acknowledgments. *ACM SIGCOMM Computer Communication Review*, 28(5):4–21, 1998. doi:10.1145/303297.303301.

5 Itai Ashlagi, Yossi Azar, Moses Charikar, Ashish Chiplunkar, Ofir Geri, Haim Kaplan, Rahul Makhijani, Yuyi Wang, and Roger Wattenhofer. Min-Cost Bipartite Perfect Matching with Delays. 81:1:1–1:20, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.1.

6 Yossi Azar, Ashish Chiplunkar, and Haim Kaplan. Polylogarithmic bounds on the competitiveness of min-cost perfect matching with delays. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1051–1061. SIAM, 2017. doi:10.1137/1.9781611974782.67.

7 Yossi Azar, Ashish Chiplunkar, Shay Kutten, and Noam Touitou. Set cover with delay - clairvoyance is not required. In *28th Annual European Symposium on Algorithms, ESA 2020, September 7-9, 2020, Pisa, Italy (Virtual Conference)*, volume 173 of *LIPIcs*, pages 8:1–8:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ESA.2020.8.

8 Yossi Azar, Arun Ganesh, Rong Ge, and Debmalya Panigrahi. Online service with delay. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 551–563, 2017. doi:10.1145/3055399.3055475.

9 Yossi Azar and Shahar Lewkowicz. Online joint replenishment problem with arbitrary holding and backlog costs. In *Proceedings of the 2026 ACM-SIAM Symposium on Discrete Algorithms, SODA 2026, Vancouver, Canada, January 11-14, 2026*, pages 2702–2725. SIAM, 2026. doi:10.1137/1.9781611978971.99.

10 Yossi Azar, Runtian Ren, and Danny Vainstein. The min-cost matching with concave delays problem. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 301–320. SIAM, 2021. doi:10.1137/1.9781611976465.20.

11 Yossi Azar and Noam Touitou. General framework for metric optimization problems with delay or with deadlines. In *Proceedings of the 60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019*, pages 60–71. IEEE Computer Society, 2019. doi:10.1109/FOCS.2019.00013.

12 Yossi Azar and Noam Touitou. Beyond tree embeddings–a deterministic framework for network design with deadlines or delay. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1368–1379. IEEE, 2020. doi:10.1109/FOCS46700.2020.00129.

13 Francis Bach. Submodular functions: from discrete to continuous domains. *Mathematical Programming*, 175(1):419–459, 2019. doi:10.1007/s10107-018-1248-6.

14 Marcin Bienkowski, Martin Böhm, Jaroslaw Byrka, Marek Chrobak, Christoph Dürr, Lukáš Folwarcznỳ, Łukasz Jeż, Jiří Sgall, Nguyen Kim Thang, and Pavel Veselỳ. Online algorithms for multilevel aggregation. *Operations Research*, 68(1):214–232, 2020. doi:10.1287/OPRE.2019.1847.

15 Marcin Bienkowski, Martin Böhm, Jaroslaw Byrka, Marek Chrobak, Christoph Dürr, Lukáš Folwarczný, Lukasz Jez, Jirí Sgall, Kim Thang Nguyen, and Pavel Veselý. New results on multi-level aggregation. *Theor. Comput. Sci.*, 861:133–143, 2021. doi:10.1016/j.tcs.2021.02.016.

16 Marcin Bienkowski, Jaroslaw Byrka, Marek Chrobak, Lukasz Jez, Dorian Nogneng, and Jirí Sgall. Better approximation bounds for the joint replenishment problem. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 42–54. SIAM, 2014. doi:10.1137/1.9781611973402.4.

17 Marcin Bienkowski, Artur Kraska, and Paweł Schmidt. A match in time saves nine: Deterministic online matching with delays. In *International Workshop on Approximation and Online Algorithms*, pages 132–146. Springer, 2017.

18 Marcin Bienkowski, Artur Kraska, and Pawel Schmidt. Online service with delay on a line. In Zvi Lotker and Boaz Patt-Shamir, editors, *Structural Information and Communication Complexity - 25th International Colloquium, SIROCCO 2018, Ma'ale HaHamisha, Israel, June 18-21, 2018, Revised Selected Papers*, Lecture Notes in Computer Science, pages 237–248. Springer, 2018. doi:10.1007/978-3-030-01325-7_22.

19 Carlos Brito, Elias Koutsoupias, and Shailesh Vaya. Competitive analysis of organization networks or multicast acknowledgement: how much to wait? In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2004, New Orleans, Louisiana, USA, January 11-14, 2004*, pages 627–635. SIAM, 2004. URL: http://dl.acm.org/citation.cfm?id=982792.982886.

20 Niv Buchbinder, Moran Feldman, Joseph (Seffi) Naor, and Ohad Talmon. O(depth)-competitive algorithm for online multi-level aggregation. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1235–1244. SIAM, 2017. doi:10.1137/1.9781611974782.80.

21 Niv Buchbinder, Tracy Kimbrel, Retsef Levi, Konstantin Makarychev, and Maxim Sviridenko. Online make-to-order joint replenishment model: Primal-dual competitive algorithms. *Oper. Res.*, 61(4):1014–1029, 2013. doi:10.1287/OPRE.2013.1188.

22 Rodrigo A. Carrasco, Kirk Pruhs, Cliff Stein, and José Verschae. The online set aggregation problem. In *LATIN 2018: Theoretical Informatics - 13th Latin American Symposium, Buenos Aires, Argentina, April 16-19, 2018, Proceedings*, volume 10807 of *Lecture Notes in Computer Science*, pages 245–259. Springer, 2018. doi:10.1007/978-3-319-77404-6_19.

23 Ryder Chen, Jahanvi Khatkar, and Seeun William Umboh. Online weighted cardinality joint replenishment problem with delay. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, Paris, France, July 4-8, 2022*, volume 229 of *LIPIcs*, pages 40:1–40:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ICALP.2022.40.

24 David D Clark. Rfc0813: Window and acknowledgement strategy in tcp, 1982.

25 Ruy De Oliveira and Torsten Braun. A dynamic adaptive acknowledgment strategy for tcp over multihop wireless networks. In *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, volume 3, pages 1863–1874. IEEE, 2005. doi:10.1109/INFCOM.2005.1498465.

26 Lindsey Deryckere and Seeun William Umboh. Online matching with set and concave delays. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2023, September 11-13, 2023, Atlanta, Georgia, USA*, volume 275 of *LIPIcs*, pages 17:1–17:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.APPROX/RANDOM.2023.17.

27 Michael Dinitz, Jeremy T. Fineman, and Seeun William Umboh. Learning-augmented online algorithms for nonclairvoyant joint replenishment problem with deadlines. In Sayan Bhattacharya, Danupon Nanongkai, Michael Benedikt, and Gabriele Puppis, editors, *53rd International Colloquium on Automata, Languages, and Programming, ICALP 2026, Royal Holloway, University of London, Egham, United Kingdom, July 7-10, 2026*, volume 374 of *LIPIcs*, pages 77:1–77:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. doi:10.4230/LIPIcs.ICALP.2026.77.

28 Daniel R Dooly, Sally A Goldman, and Stephen D Scott. Tcp dynamic acknowledgment delay (extended abstract) theory and practice. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 389–398, 1998.

29 Daniel R Dooly, Sally A Goldman, and Stephen D Scott. On-line analysis of the tcp acknowledgment delay problem. *Journal of the ACM (JACM)*, 48(2):243–273, 2001. doi:10.1145/375827.375843.

30 Marc Dufay and Roger Wattenhofer. A deterministic polylogarithmic competitive algorithm for matching with delays. In Kasper Green Larsen and Barna Saha, editors, *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2026, Vancouver, BC, Canada, January 11-14, 2026*, pages 3936–3964. SIAM, 2026. doi:10.1137/1.9781611978971.144.

31 Yuval Emek, Shay Kutten, and Roger Wattenhofer. Online matching: haste makes waste! In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 333–344, 2016. doi:10.1145/2897518.2897557.

32 Yuval Emek, Yaacov Shapiro, and Yuyi Wang. Minimum cost perfect matching with delays for two sources. *Theor. Comput. Sci.*, 754:122–129, 2019. doi:10.1016/J.TCS.2018.07.004.

33 Tomer Ezra, Stefano Leonardi, Michal Pawlowski, Matteo Russo, and Seeun William Umboh. Universal optimization for non-clairvoyant subadditive joint replenishment. *Algorithmica*, 88(3):43, 2026. doi:10.1007/S00453-026-01389-1.

34 Junhao Gan, Xiao Sun, and Seeun William Umboh. Online matching with size-based and convex delays. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2026*, 2026. To appear.

35 Anupam Gupta, Amit Kumar, and Debmalya Panigrahi. A hitting set relaxation for k-server and an extension to time-windows. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 504–515. IEEE, 2021. doi:10.1109/FOCS52979.2021.00057.

36 Anupam Gupta, Amit Kumar, and Debmalya Panigrahi. Caching with time windows and delays. *SIAM J. Comput.*, 51(4):975–1017, 2022. doi:10.1137/20M1346286.

37 Péter Györgyi, Tamás Kis, Tímea Tamási, and József Békési. Joint replenishment meets scheduling. *Journal of Scheduling*, 26(1):77–94, 2023. doi:10.1007/S10951-022-00768-0.

38 Kun He, Sizhe Li, Enze Sun, Yuyi Wang, Roger Wattenhofer, and Weihao Zhu. Randomized algorithm for MPMD on two sources. In Jugal Garg, Max Klimm, and Yuqing Kong, editors, *Web and Internet Economics - 19th International Conference, WINE 2023, Shanghai, China, December 4-8, 2023, Proceedings*, Lecture Notes in Computer Science, pages 348–365. Springer, 2023. doi:10.1007/978-3-031-48974-7_20.

39 Anna R. Karlin, Claire Kenyon, and Dana Randall. Dynamic TCP Acknowledgment and Other Stories about e/(e - 1). *Algorithmica*, 36(3):209–224, July 2003. doi:10.1007/s00453-003-1013-x.

40 Anna R. Karlin, Mark S. Manasse, Larry Rudolph, and Daniel Dominic Sleator. Competitive snoopy caching. *Algorithmica*, 3:77–119, 1988. doi:10.1007/BF01762111.

41 Predrag Krnetic, Darya Melnyk, Yuyi Wang, and Roger Wattenhofer. The k-server problem with delays on the uniform metric space. In Yixin Cao, Siu-Wing Cheng, and Minming Li, editors, *31st International Symposium on Algorithms and Computation, ISAAC 2020, Hong Kong (Virtual Conference), December 14-18, 2020*, LIPIcs, pages 61:1–61:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ISAAC.2020.61.

42 Ngoc Mai Le, Seeun William Umboh, and Ningyuan Xie. The power of clairvoyance for multi-level aggregation and set cover with delay. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1594–1610. SIAM, 2023. doi:10.1137/1.9781611977554.CH59.

43 Mathieu Mari, Michal Pawlowski, Runtian Ren, and Piotr Sankowski. Online matching with delays and stochastic arrival times. *Theory Comput. Syst.*, 69(1):12, 2025. doi:10.1007/S00224-024-10207-6.

44 Adam Meyerson. The parking permit problem. In *46th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2005, Pittsburgh, PA, USA, October 23-25, 2005, Proceedings*, pages 274–284. IEEE Computer Society, 2005. doi:10.1109/SFCS.2005.72.

45 Benjamin Moseley, Aidin Niaparast, and R Ravi. Putting off the catching up: Online joint replenishment problem with holding and backlog costs. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3865–3883. SIAM, 2025. doi:10.1137/1.9781611978322.130.

46 Joseph (Seffi) Naor, Seeun William Umboh, and David P. Williamson. Tight bounds for online weighted tree augmentation. *Algorithmica*, 84(2):304–324, 2022. doi:10.1007/S00453-021-00888-7.

47 Kalen Patton, Matteo Russo, and Sahil Singla. Submodular norms with applications to online facility location and stochastic probing. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2023, September 11-13, 2023, Atlanta, Georgia, USA*, volume 275 of *LIPIcs*, pages 23:1–23:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.APPROX/RANDOM.2023.23.

48 Steven S. Seiden. A guessing game and randomized online algorithms. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 592–601. ACM, 2000. doi:10.1145/335305.335385.

49 David Shmoys, Varun Suriyanarayana, and Seeun William Umboh. Improved online algorithms for inventory management problems with holding and delay costs: Riding the wave makes things simpler, stronger, & more general. In *Proceedings of the 2026 ACM-SIAM Symposium on Discrete Algorithms, SODA 2026, Vancouver, Canada, January 11-14, 2026*, pages 2726–2742. SIAM, 2026. doi:10.1137/1.9781611978971.100.

50 Noam Touitou. Nearly-tight lower bounds for set cover and network design with deadlines/delay. In *32nd International Symposium on Algorithms and Computation, ISAAC 2021, December 6-8, 2021, Fukuoka, Japan*, volume 212 of *LIPIcs*, pages 53:1–53:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ISAAC.2021.53.

51 Noam Touitou. Frameworks for nonclairvoyant network design with deadlines or delay. In *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPIcs*, pages 105:1–105:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ICALP.2023.105.

52 Noam Touitou. Improved and deterministic online service with deadlines or delay. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 761–774. ACM, 2023. doi:10.1145/3564246.3585107.

53 Noam Touitou. Nearly-optimal algorithm for non-clairvoyant service with delay. In Olaf Beyersdorff, Michal Pilipczuk, Elaine Pimentel, and Kim Thang Nguyen, editors, *42nd International Symposium on Theoretical Aspects of Computer Science, STACS 2025, Jena, Germany, March 4-7, 2025*, LIPIcs, pages 74:1–74:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.STACS.2025.74.

Approximation Algorithms for Discounted Graph Search with Norm Objectives

Svenja M. Griesbach
Department of Computer Science, RWTH Aachen University, Germany

Felix Hommelsheim
Department of Computer Science, University of Cologne, Germany

Max Klimm
Institute of Mathematics, Technische Universität Berlin, Germany

Abstract

We introduce a unified framework for classical search and routing problems, including pathwise search, expanding search, the minimum spanning tree problem, and the traveling salesperson problem. The framework is based on two parameters. The first is a discount factor $\alpha \in [0, 1]$: the first traversal of an edge incurs its full cost, whereas each subsequent traversal incurs only an α-fraction of this cost. For a path starting at a designated root vertex, the α-latency of a vertex is the discounted cost accumulated until the vertex is first visited. The second parameter is a norm parameter $p \geq 1$. The objective is to find a root-starting path that visits all vertices and minimizes the p-norm of the resulting vector of α-latencies.

The model interpolates between several well-studied objectives. For $p = 1$ and $\alpha = 1$, it recovers pathwise search; for $p = 1$ and $\alpha = 0$, it recovers expanding search. As p tends to infinity, the objective converges to a makespan-type criterion. At the endpoints $\alpha = 1$ and $\alpha = 0$, this limiting objective corresponds to TSP-type and MST-type behavior, respectively. For $p = 1$, we give polynomial-time constant-factor approximation algorithms for all $\alpha \in [0, 1]$, matching the best known guarantees for expanding search at $\alpha = 0$ and pathwise search at $\alpha = 1$. For general $p \geq 1$, we obtain a randomized constant-factor approximation algorithm and a derandomized pseudo-polynomial-time algorithm with the same guarantee.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases Approximation Algorithm, Expanding Search, Pathwise Search, Search Problem, Graph Exploration, Traveling Repairperson Problem, Minimum Latency Problem

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.27

Category APPROX

Related Version *Full Version*: `https://arxiv.org/abs/2607.11301`

Funding The work of the first author is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) by the grant Ho 3831/9-1 (project ID: 514505843). The work of the second author is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) by the grant HO 7562/2-1 (project ID: 573939419). The work of the third author is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany´s Excellence Strategy – The Berlin Mathematics Research Center MATH+ (EXC-2046/1, EXC-2046/2, project ID: 390685689).

Acknowledgements We thank Kevin Schewior for fruitful discussions.

1 Introduction

Pathwise search, expanding search, the traveling salesperson problem, and the minimum spanning tree problem are four fundamental models for exploring or connecting a network. At first glance, these problems optimize rather different objectives: pathwise and expanding


© Svenja M. Griesbach, Felix Hommelsheim, and Max Klimm;
licensed under Creative Commons License CC-BY 4.0

search minimize sums of discovery times, while the traveling salesperson problem and the minimum spanning tree problem minimize the time until all vertices are reached or connected. In this paper, we study a common framework that unifies these problems through two parameters. The first parameter determines how repeated traversals of edges are charged, and the second parameter determines how the individual vertex latencies are aggregated.

We are given an undirected graph $G = (V, E)$ with non-negative edge cost $c_e \in \mathbb{N}$, and a designated start vertex s. A solution is a traversal of the graph, i.e., a path starting in s that may visit edges more than once and eventually visits all vertices. For a vertex v, its latency is the time at which v is visited for the first time. Classical search problems ask for a traversal that minimizes the sum of these latencies.

The *pathwise search problem* asks for such a traversal when every traversal of an edge e requires c_e time units. Thus, the latency of a vertex is equal to the total cost of the prefix of the path until the vertex is first visited, and the goal is to minimize the sum of the latencies of all vertices. The problem, also known as the *traveling repairperson problem*, captures situations in which all traversals of an edge take the same amount of time. Hence, it has been used as a model for the movement of a repairperson in a road network or for disk heads on a hard drive. The problem is NP-hard even on weighted trees (Sitters [27]), so much research has focused on approximation algorithms. An algorithm is a ϱ-approximation algorithm if for any instance it runs in polynomial time and the cost of the solution output by the algorithm is at most $\varrho \cdot \textsc{Opt}$, where Opt denotes the cost of an optimum solution to the respective instance. The factor ϱ is called the approximation ratio or guarantee. The best currently known approximation algorithm for general graphs with unit weights for all vertices yields a 3.59-approximation (Chaudhuri et al. [15]).

The *expanding search problem* asks for a sequence of edges with the property that every prefix of edges is a connected subgraph and, without loss of generality, a tree containing s. The latency of a vertex is the total cost of the edges added until the vertex first appears in the sequence, and the goal is again to minimize the sum of the latencies of all vertices. It captures situations in which a tree network needs to be installed, and the cost of an edge is interpreted as the time needed to establish a connection between its end vertices. Hence, it has been used as a model for clearing paths in an area devastated by disasters or for mining. The problem is NP-hard (Averbakh and Pereira [9]) and the best currently known approximation algorithm for general graphs yields a 5.44-approximation (Griesbach et al. [22]).

We propose and study a general model of graph exploration that we term the *discounted graph search problem*. As in pathwise search, a solution is a traversal π starting in s that may visit edges more than once and eventually visits all vertices. The first parameter of the model is a discount factor $\alpha \in [0, 1]$. While the first traversal of an edge e in the sequence requires c_e time units, every further traversal of the same edge requires only $\alpha \cdot c_e$ time units. We call the cost of a path where repeated traversals are discounted in this way the *α-cost* of the path. Given a traversal π, the *α-latency* $C_{\alpha,v}(\pi)$ of a vertex v is defined as the α-cost of the smallest prefix of π that visits v.

The second parameter is a norm parameter $p \geq 1$, which determines how the individual α-latencies are aggregated. For a traversal π, its p-norm α-latency is

$$\|\mathbf{C}_\alpha(\pi)\|_p = \left(\sum_{v \in V} C_{\alpha,v}(\pi)^p \right)^{1/p}.$$

The goal is to compute a traversal π minimizing this quantity.

This two-parameter model contains several classical problems as special cases or limiting cases. For $p = 1$ and $\alpha = 1$, every traversal of an edge is charged its full cost, and the objective is the sum of the first-visit times of all vertices. Thus, we recover the pathwise search problem. For $p = 1$ and $\alpha = 0$, repeated traversals of already used edges are free. Therefore, only the cost of newly added edges contributes to the discovery time of vertices, and we recover the expanding search problem.

The parameter p interpolates between sum-of-latencies objectives and makespan-type objectives. As $p \to \infty$, the p-norm objective converges to the maximum α-latency of any vertex. For $\alpha = 1$, this limiting objective asks for a shortest traversal starting at s that visits all vertices, since the maximum latency is exactly the time when the last vertex is first reached. This is the path version of the traveling salesperson problem. For $\alpha = 0$, the maximum α-latency is the total cost of the distinct edges that have been introduced by the time all vertices are reached. Minimizing this quantity is therefore equivalent to finding a minimum-cost connected subgraph spanning all vertices, and hence to the minimum spanning tree problem. Intermediate values of $\alpha \in (0, 1)$ capture situations where the first traversal of an edge is more time-consuming than later traversals, for example, because of additional delays due to pathfinding or clearing a road, while further traversals still require a non-negligible amount of time. Intermediate values of p capture settings in which one wants to balance the average discovery time of vertices with the time until the last vertices are reached. The parameter p can also be interpreted from a fairness perspective. For $p = 1$, the objective minimizes the total latency and thus corresponds to a social-welfare objective, whereas larger values of p put increasing emphasis on vertices with large latency; in the limit $p \to \infty$, the objective becomes an egalitarian objective that minimizes the worst latency. The special case of $p = 2$ and $\alpha = 1$ has been studied as the *traveling firefighter problem* in the literature [16]. The authors motivate this particular choice of objective by firefighters looking for an order in which to tackle wildfires whose damage grows quadratically with the elapsed time.

1.1 Our Results

We first note that the discounted graph search problem is computationally hard already in the case $p = 1$. In particular, the hardness of pathwise and expanding search carries over to the endpoints $\alpha = 1$ and $\alpha = 0$, and we also show that the problem remains hard for every fixed intermediate value $\alpha \in (0, 1)$. The proof can be found in the full version of the paper.

▶ **Theorem 1.** *For $p = 1$ and every constant $\alpha \in [0, 1]$, there exists a constant $\varepsilon > 0$ such that there is no polynomial-time $(1 + \varepsilon)$-approximation algorithm for the discounted graph search problem with discount factor α, unless* P = NP.

We therefore focus on approximation algorithms. We first consider the case $p = 1$, where the objective is the total α-latency. For this case, we obtain a polynomial-time approximation algorithm with approximation ratio

$$\varrho_1(\alpha) \coloneqq \begin{cases} 2\mathrm{e} & \text{if } \alpha = 0, \\ 2\frac{\alpha}{(1+\alpha)W(\alpha/\mathrm{e})} & \text{otherwise,} \end{cases}$$

where $W : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is the Lambert-W function, which assigns $x \geq 0$ the unique value $w \geq 0$ such that $w\mathrm{e}^w = x$. This gives the following result.

▶ **Theorem 2.** *For $p = 1$ there is a polynomial-time $\varrho_1(\alpha)$-approximation algorithm for the discounted graph search problem.*

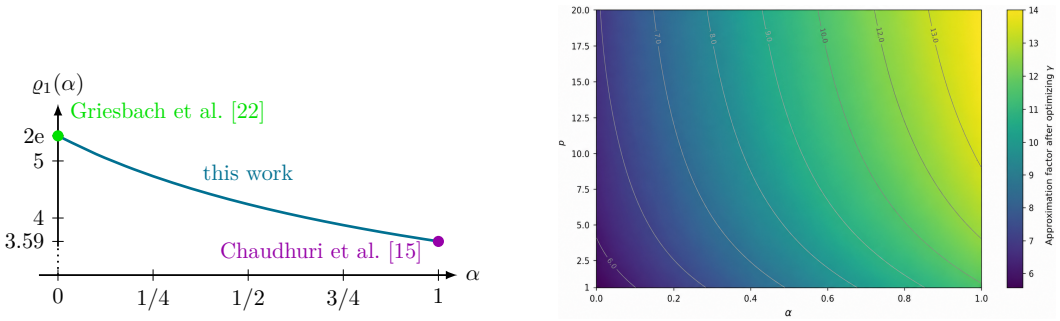


Figure 1 Approximation ratios $\varrho(\alpha)$ obtained in this work: (a) for $p = 1$ as a function of α; (b) as a function of $p > 1$ and α.

For $\alpha = 0$, the approximation factor is 2e, which matches the best currently known approximation guarantee for the expanding search problem (Griesbach et al. [22]). For $\alpha = 1$, the approximation factor is 3.59, which matches the best currently known approximation guarantee for the pathwise search problem in general graphs with unit vertex weights (Chaudhuri et al. [15]). For arbitrary values of $\alpha \in [0, 1]$, the approximation ratio interpolates smoothly between these two values; see Figure 1.

Like the algorithms of Chaudhuri et al. [15] and Griesbach et al. [22], our algorithm is based on a sequence of certain trees of exponentially growing cost. The trees that we consider are related to the concept of *good* trees used by Chaudhuri et al. [15] in their 3.59-approximation for the pathwise search problem. Roughly speaking, a tree is called *good* if it contains a certain number of vertices and its cost is bounded by the cost of an appropriate path visiting the same number of vertices. We extend this concept to general values of α. Specifically, we define good k-trees for α and show that such trees can be computed in polynomial time for the values of k required by our algorithm. For $\alpha = 1$, this recovers the trees used by Chaudhuri et al. [15]. For $\alpha = 0$, the good k-trees correspond to k-MSTs, and the resulting bound matches the best known approximation guarantee for expanding search.

We then turn to the more general case $p \geq 1$, where the objective is the p-norm of the vertices' α-latencies. For $p > 1$, the approach for the case $p = 1$ cannot be applied directly. The main difficulty is that the objective is no longer linear in the latencies, and hence the auxiliary-graph construction used for $p = 1$ no longer directly captures the contribution of each additional tree. Moreover, the good-tree machinery from the $p = 1$ case does not provide suitable trees for every value of k.

We therefore use a randomized construction that is closer in spirit to the geometric tree sequences used for $p = 1$, but relies only on 2-approximate k-MSTs. For every $k \in [n]$, we compute such a tree, choose a random geometric offset, and concatenate trees whose costs grow geometrically. Analyzing the p-th moments of the resulting vertex latencies gives the following guarantee.

▶ **Theorem 3.** *For every $p \geq 1$, there is a polynomial-time randomized algorithm for the discounted graph search problem with approximation factor*

$$\varrho_p(\alpha) = \min_{\gamma > 1} \left\{ \frac{2(1+\alpha)}{\gamma - 1} \left[\frac{\gamma^{2p} - \gamma^p}{p \ln \gamma} \right]^{1/p} \right\}.$$

The approximation factor from Theorem 3 is depicted in Figure 1. For fixed α, the optimized factor tends to $8(1 + \alpha)$ as $p \to \infty$. Over the range $\alpha \in [0, 1]$, the smallest value is attained at $\alpha = 0$ and $p = 1$, where the factor is 2e, while the largest value is approached for $\alpha = 1$ and $p \to \infty$, where the factor tends to 16.

For the traveling firefighter problem, corresponding to the cases $\alpha = 1$ and $p = 2$, this yields an approximation factor of ≈ 11.28. This is worse by a factor of 2 than the approximation factor of ≈ 5.65 claimed by Farhadi et al. [16]. However, as we discuss in Section 1.2, we do not believe that this discrepancy can be resolved straightforwardly.

Finally, we derandomize the p-norm algorithm using an auxiliary graph, as in the case $p = 1$. The difference is that for $p > 1$ the state must also record the accumulated cost of the trees traversed so far, since this quantity determines the nonlinear contribution to the p-norm objective. This yields a two-dimensional, time-expanded auxiliary graph and the following deterministic guarantee.

▶ **Theorem 4.** *For every $p \geq 1$, there is a deterministic pseudo-polynomial-time algorithm for the discounted graph search problem with approximation factor $\varrho_p(\alpha)$ defined in Theorem 3. Moreover, if the tree costs are integral and polynomially bounded, then the algorithm runs in polynomial time.*

1.2 Related Work

The pathwise search problem is also known as the *traveling repairperson problem*. This problem was shown to be NP-hard on general graphs by Sahni and Gonzalez [26], and to be NP-hard even on weighted trees by Sitters [27]. In the weighted setting, each vertex has a non-negative weight and the task is to minimize the weighted sum of latencies. The problem can be solved efficiently on unweighted trees and paths [1, 17, 25]. The first approximation algorithm for this problem was devised by Blum et al. [12] who gave a 144-approximation for general graphs and an 8-approximation for weighted trees. Goemans and Kleinberg [20] improved the approximation ratio for general graphs to 21.55 and the one for weighted trees to 3.59. In particular, they show that a β-approximation for the rooted k-MST problem of finding a shortest tree containing a given root and $k-1$ further vertices implies an approximation ratio of 3.59β for the traveling repairperson problem. Thus, the improved k-MST approximations of $\beta = 3$ by Garg [18], of $\beta = 2.5$ by Arya and Ramesh [8], of $\beta = 2 + \varepsilon$ by Arora and Karakostas [7], and of $\beta = 2$ by Garg [19] immediately yield better approximations of 10.77, 8.98, $7.18 + \varepsilon$, and 7.18, respectively. In addition, the work of Arora [5] implies an improved approximation ratio of $3.59 + \varepsilon$ for the Euclidean case. Archer et al. [4] showed that the approximation ratio of $7.18 + \varepsilon$ can be obtained with fewer calls to the k-MST subroutine. Chaudhuri et al. [15] improved the approximation ratio for general graphs to match the ratio of 3.59 for weighted trees. Archer and Blasiak [3] further improved the approximation ratio for weighted trees to 3.03. Arora and Karakostas [6] gave a quasi-polynomial-time approximation scheme (QPTAS) both for weighted trees and for Euclidean instances. Sitters [28] gave a polynomial-time approximation scheme (PTAS) for Euclidean instances, weighted trees, and planar graphs.

The expanding search problem has been shown to be NP-hard by Averbakh and Pereira [9]. Alpern and Lidbetter [2] proposed an algorithm for the expanding search problem on weighted trees. The first approximation algorithm was devised by Hermans et al. [23] who gave an 8-approximation. The approximation ratio was improved by Griesbach et al. [22] to $2\mathrm{e} \approx 5.44$.

Farhadi et al. [16] study the L_p-TSP, where the objective is to minimize the p-norm of the vector of visit times; this problem interpolates between the traveling repairperson problem for $p = 1$ and the path variant of TSP for $p \to \infty$ and corresponds to our case of $\alpha = 1$. They claimed to have a universal 8-approximation that is valid for all p simultaneously. If correct, this would improve over a 16-approximation by Golovin et al. [21]. Farhadi et al. [16] further claimed a 5.65-approximation for $p = 2$. However, we believe that both their results have a

substantial gap. Their algorithms rely on the ability to compute good k-trees for all values $k \in [n]$; Lemma 3.1 in their paper cites Chaudhuri et al. [15] for the claim that this can be done in polynomial time. However, this is not what Chaudhuri et al. [15] show. They only show "how to obtain such k-trees for some values of k" and "that it actually suffices to use the few k-trees that we found" [15, p. 3]. The algorithms of Farhadi et al. [16], on the other hand, crucially require good k-trees for *all* values of k to be computable in polynomial time. It seems to be an open problem whether this can be done. A positive answer to this open problem appears to require new techniques, as Garg [19, p. 1] writes that "we cannot argue that the cost of the tree picked is at most the cost of the best k-stroll. As was shown in [15] such an argument can be made for certain values of k by exploiting the same slack in the Goemans-Williamson argument". Our algorithms for $p > 1$ avoid this issue by computing 2-approximate k-MSTs for all values of k instead of good k-trees; the former can be done with the algorithm of Garg [19]. This is exactly why we incur an additional loss of a factor of 2. The discussion of Garg cited above makes it plausible that this additional factor of 2 cannot be avoided with current techniques. This route was also taken by Golovin et al. [21], who use 2-approximate k-MSTs for all values of k instead of good k-trees. It is interesting to note that our approximation guarantees for any $p \geq 1$ and $\alpha \in [0, 1]$ are strictly below 16 and approach this value when $\alpha = 1$ and $p \to \infty$.

Norm-based interpolation also appears in ordered optimization problems such as ordered k-median, which interpolates between k-median and k-center [13]. In scheduling and load balancing, one often minimizes the L_p-norm of the machine-load vector, which similarly interpolates between average-load objectives and makespan minimization; see, e.g., the work of Awerbuch et al. [10] and subsequent work on all-norm and minimum-norm load balancing [11, 14, 24]. While these problems differ from ours, they illustrate the role of p-norm objectives as a natural way to interpolate between sum-type and bottleneck-type criteria.

2 Preliminaries

Let $G = (V, E)$ be an undirected graph with $|V| = n + 1$ and a designated start vertex s and let $\alpha \in [0, 1]$ be the discount factor. Each edge $e \in E$ has a non-negative cost $c_e \in \mathbb{N}$. We assume that $c_e > 0$ for every edge incident to s since otherwise these edges can be contracted. We call a sequence of edges $\pi = (e_1, \ldots, e_k)$ a *path* if for all $1 \leq i < k$ the end vertex of e_i and the start vertex of e_{i+1} coincide and the sequence starts in s. If the path also ends in s, we call it a *tour*. The set of all tours is denoted by Π. For two paths $\pi = (e_1, \ldots, e_k)$ and $\pi' = (e'_1, \ldots, e'_l)$ where the end vertex of π coincides with the start vertex of π', we write $\pi + \pi' = (e_1, \ldots, e_k, e'_1, \ldots, e'_l)$ for the concatenation of those two paths.

For a sequence of edges π, we define the *α-cost* $c_\alpha(\pi)$ of π as the sum of all edges in π, where for the second and further traversals of an edge e its cost c_e is multiplied by α. We call a sequence π a *tree* if the set of edges in π spans a tree in G that contains s. Since a tree contains no edge more than once, the α-cost of a tree is independent of α, and we simply write $c(\pi)$. The following lemma states the simple fact that every tree can be transformed into a tour by increasing the α-cost by at most a factor of $1 + \alpha$. We state it here for future reference; its proof can be found in the full version of the paper.

▶ **Proposition 5.** *For a tree T, there exists a tour $\pi(T)$ that visits all vertices in T and has α-cost at most $c_\alpha(\pi(T)) \leq (1 + \alpha)c(T)$.*

For a vertex $v \in V$ visited by a tour $\pi \in \Pi$, we denote by π_v the prefix of π until v is visited for the first time. The *α-latency* $C_{\alpha,v}(\pi)$ of vertex v in π is defined as $C_{\alpha,v}(\pi) := c_\alpha(\pi_v)$ where we set $C_{\alpha,v}(\pi) = \infty$ if v is not visited by π at all. Note that $C_{\alpha,s}(\pi) = 0$. The *total*

α-*latency* $C_\alpha(\pi)$ of π is then defined as $C_\alpha(\pi) \coloneqq \sum_{v \in V} C_{\alpha,v}(\pi)$. In particular, we have $C_\alpha(\pi) = \infty$ if there is a vertex $v \in V$ that is never visited by π. The set of feasible solutions for the *discounted graph search problem* is the set Π. Further, a tour $\pi^* \in \Pi$ is optimal for the discounted graph search problem with $\alpha \in [0,1]$ if $C_\alpha(\pi^*) \leq C_\alpha(\pi)$ for all tours $\pi \in \Pi$.

A k-*tree*, k-*path*, or k-*tour* is a tree, path, or tour, respectively, that contains s and visits at least $k+1$ distinct vertices of V. For a k-tour $\pi \in \Pi$, we denote by π_k the prefix of π until the k-th distinct vertex of $V \setminus \{s\}$ is visited for the first time. A k-path π^* is *optimal* with respect to α if $c_\alpha(\pi^*) \leq c_\alpha(\pi)$ for all k-paths π. If α is clear from context, we simply say that π^* is optimal. Further, a k-tree T is called *good* if its cost is at most $c(T) \leq \frac{2}{1+\alpha} c_\alpha(\pi^*)$, where π^* is an optimal k-path.

3 Algorithms for the 1-Norm of α-Latencies

Recall that $\varrho(\alpha) = 2\frac{\alpha}{(1+\alpha)W(\alpha/\mathrm{e})}$ when $\alpha \in (0,1]$ and $\varrho(0) = 2\mathrm{e}$. We prove the following result.

▶ **Theorem 2.** *For $p = 1$ there is a polynomial-time $\varrho_1(\alpha)$-approximation algorithm for the discounted graph search problem.*

The intuitive idea behind the algorithm is to concatenate a specific subset of *good trees*. We construct an initial set of good trees in Section 3.1 using an algorithm by Chaudhuri et al. [15]. They introduced this algorithm for the computation of good k-trees for a specific subset of values of k and in the special case of $\alpha = 1$. A more thorough analysis of their approach yields that the obtained trees are indeed good trees for arbitrary choices of $\alpha \in [0,1]$. Since the algorithm does not return good k-trees for all values of $k \in [n]_0$, we use so-called phantom trees for the remaining values of k; the details of this construction can be found in the full version of the paper. Next, we introduce the concatenating algorithm in Section 3.2, which takes as input a set of good trees and returns a feasible solution to the discounted graph search problem. More precisely, it constructs an auxiliary graph H whose vertices correspond to the good trees. Within this auxiliary graph, the algorithm computes a shortest path and concatenates the corresponding trees to the desired tour. The algorithm is thus well-defined only if the input contains only real trees. We temporarily ignore this constraint and show in Section 3.3 that, under this relaxation, the returned (phantom) tour has an approximation guarantee of $\varrho_1(\alpha)$. The analysis builds on the construction of a randomized path in the auxiliary graph that visits exponentially growing trees. Then, we show that the same guarantee can be obtained by a path that visits only real trees. This allows us to restrict the input to good real trees while maintaining the approximation guarantee. Combining these results yields a well-defined algorithm that returns the desired solution. The running-time analysis then concludes the proof of Theorem 2.

3.1 Finding Good Trees

To obtain good k-trees, we use an algorithm by Chaudhuri et al. [15]. They study the pathwise search problem, i.e., the discounted graph search problem with $p = \alpha = 1$, for which they introduce a primal–dual algorithm that computes good k-trees for a subset of values of k. With a more thorough analysis, we can show that these trees are indeed good trees for all choices of $\alpha \in [0,1]$.

A subtle issue is that the primal–dual procedure is not monotone in the vertex-budget parameter. In contrast to what is sometimes assumed in related Lagrangian approaches for k-MST [7], increasing the parameter need not produce a larger tree, even before the final pruning step. This is because the parameter can change the order in which moats merge and become inactive. An explicit example can be found in the full version of the paper.

A detailed explanation of how the framework of Chaudhuri et al. [15] needs to be adapted to compute good k-trees for all values of $k \in [n]$ can be found in the full version of the paper. For this purpose, it is important to distinguish between values of k for which a good k-tree can be computed and those for which this is not the case. In the latter case, we define a phantom tree, which is not an actual tree but a placeholder whose costs are interpolated from the costs of the closest two real k-trees. We later argue that these phantom trees are not used by our approximation algorithm, so their inclusion as placeholders does not introduce any issues down the line.

The main result needed here is the following lemma; its proof can be found in the full version of the paper.

▶ **Lemma 6.** *There is a polynomial-time algorithm that computes a set $\mathcal{T}$ containing, for each value $k \in [n]_0$, a real or phantom tree $T_k \in \mathcal{T}$. Each such T_k is a good k-tree.*

3.2 The Concatenating Algorithm

We introduce the *concatenating algorithm*, which takes as input a set of good k-trees, selects a subset of these, and concatenates them to obtain a solution for the discounted graph search problem. To this end, let $\mathcal{I}$ be a set of good k-trees for a subset of values $k \in [n]_0$ and let $I \subseteq [n]_0$ be such that $k \in I$ if and only if $\mathcal{I}$ contains a good k-tree denoted by T_k. Then the concatenating algorithm with input $\mathcal{I}$ is defined as follows:

1. Construct a directed auxiliary graph $H = (V_H, A_H)$ with vertices $V_H = I$, edges $A_H = \{(i,j) : i < j\}$, and edge lengths $\ell_{i,j} = \big[(1+\alpha)n - \alpha j - i\big] c(T_j)$.
2. Compute a shortest 0–n-path $P^* = (n_0, n_1, \dots, n_l)$ with $n_0 = 0$ and $n_l = n$ in H.
3. Start with the empty sequence $\pi_{\text{ALG}} = (\cdot)$.
   For each phase $j = 1, \dots, l$, construct the tour $\pi(T_{n_j})$ according to Proposition 5. Let π_{f} and π_{b} be the tour obtained from traversing the sequence $\pi(T_{n_j})$ in forward or backward direction, respectively. Let $V_j \coloneqq V_{n_j} \setminus \big(\bigcup_{i=1}^{j-1} V_{n_i}\big)$ be the set of vertices that are contained in T_{n_j} but in no previous tree T_{n_i} for $1 \le i < j$. Let $\pi_{\min} \in \{\pi_{\text{f}}, \pi_{\text{b}}\}$ be such that $\sum_{v \in V_j} C_{\alpha,v}(\pi_{\text{ALG}} + \pi_{\min})$ is minimized. Set $\pi_{\text{ALG}} \leftarrow \pi_{\text{ALG}} + \pi_{\min}$.

Note that the concatenating algorithm is only well-defined under the following two assumptions. First, the input must contain a good 0-tree and a good n-tree. Second, all trees $T_{n_0}, \dots, T_{n_l}$ on the shortest path must be real trees, because otherwise Step 3 is not well defined. The second constraint is trivially fulfilled if the input $\mathcal{I}$ only contains real trees.

3.3 The Concatenating Algorithm with Phantom Trees

We want to apply the concatenating algorithm to the set $\mathcal{T}$ of good trees whose existence is guaranteed by Lemma 6. However, this set may contain phantom trees, so the concatenating algorithm is not necessarily well-defined on this input. We ignore this issue for now by assuming that all phantom trees are real trees and can therefore be transformed into tours. Under this assumption, we analyze the approximation ratio of the resulting tour π_{ALG} and then show how to obtain the same guarantee when restricting to real trees.

To this end, let $\mathcal{T}$ be the set of good k-trees whose existence is guaranteed by Lemma 6. Note that $\mathcal{T}$ contains a good (real or phantom) k-tree *for all* values of $k \in [n]_0$ and that the trees T_0 and T_n are real trees. Before applying the concatenating algorithm to $\mathcal{T}$, we slightly adjust the cost of the phantom trees $\mathcal{P} \subset \mathcal{T}$. More precisely, for a phantom tree $T_k \in \mathcal{P}$, let $k_l < k$ be maximal and $k < k_r$ be minimal such that $T_{k_l}, T_{k_r} \in \mathcal{T}$ are two real trees. We redefine the cost of T_k to

$$c(T_k) \coloneqq (1-\mu)c(T_{k_l}) + \mu\, c(T_{k_r}) \quad \text{with} \quad \mu \coloneqq \frac{k - k_l}{k_r - k_l}. \tag{1}$$

Since the phantom tree T_k was originally constructed by a linear interpolation of two real trees, this redefinition of the cost can only decrease its cost. See also [15] for a more detailed discussion. Hence, the phantom tree T_k remains a good k-tree. We denote the set $\mathcal{T}$ with the adjusted costs of the phantom trees by $\mathcal{T}'$. We analyze the approximation guarantee of the tour π_{ALG} obtained by running the concatenating algorithm on input $\mathcal{T}'$. In this section, we prove the following lemma.

▶ **Lemma 7.** *If all trees in $\mathcal{T}'$ were real trees, the tour π_{ALG} obtained from the concatenating algorithm with input $\mathcal{T}'$ is a feasible solution for the discounted graph search problem with approximation guarantee $\varrho_1(\alpha)$.*

For the remainder of this section, we denote by P^* the shortest 0–n-path in H computed in Step 2 of the concatenating algorithm on input $\mathcal{T}'$. Since each vertex $i \in V_H$ corresponds to a unique tree $T_i \in \mathcal{T}'$, we often refer to the vertices on P^* as the chosen trees. Furthermore, we extend the procedure of Step 3 to an arbitrary 0–n-path $P = (n_0, n_1, \dots, n_{l'})$ in H. More precisely, we introduce a randomized variant of Step 3 that runs as follows. We start with the empty sequence $\pi_P = (\cdot)$. For each phase $j = 1, \dots, l'$, we construct the tour $\pi(T_{n_j})$ according to Proposition 5. Then we pick a traversal direction (forward or backward) of the tour $\pi(T_{n_j})$ uniformly at random and set $\pi_P \leftarrow \pi_P + \pi(T_{n_j})$.

We proceed to analyze the sequence π_{ALG}. First, observe that P^* contains vertex n. Thus, the corresponding good n-tree T_n is traversed and appended to π_{ALG} in Step 3. This ensures that the returned tour π_{ALG} is a feasible solution to the discounted graph search problem with finite total α-latency, i.e., π_{ALG} visits all vertices. It remains to analyze the approximation factor obtained by π_{ALG}. To this end, we first show that for any given 0–n-path P in H, the expected total α-latency $\mathbb{E}[C_\alpha(\pi_P)]$ of π_P is at most the length $\ell(P)$ of P in H. Next, we use a probabilistic argument to prove the existence of a 0–n-path in H whose length is at most $\varrho_1(\alpha)$ times the total α-latency of an optimal tour. Finally, we explain how these two results imply that the concatenating algorithm on input $\mathcal{T}'$ indeed computes a tour with an approximation ratio of $\varrho_1(\alpha)$.

In that direction, we introduce the function ψ_P such that $\psi_P(k)$ gives an upper bound on the latency of the k-th distinct vertex in $V \setminus \{s\}$ visited by the tour constructed from path P in H. More formally, for a fixed 0–n-path $P = (n_0, n_1, \dots, n_l)$ in H, the function $\psi_P : [n]_0 \to \mathbb{R}_{\geq 0}$ is defined by

$$\psi_P(k) \coloneqq \begin{cases} 0 & \text{if } k = 0 \\ c(T_{n_j}) + (1+\alpha)\sum_{i=0}^{j-1} c(T_{n_i}) & \text{if } n_{j-1} < k \leq n_j, \text{for some } j \in \{1, \dots, l\}. \end{cases}$$

This definition is well-defined as $n_0 = 0$ and $n_l = n$. We obtain the following lemma. Its proof can be found in the full version of the paper.

▶ **Lemma 8.** *Let π_P be the randomized tour obtained from a 0–n-path P in H and let $v \in V$ be the k-th distinct vertex of $V \setminus \{s\}$ in π_P for some $k \in [n]_0$. Then we have $\mathbb{E}[C_{\alpha,v}(\pi_P)] \leq \psi_P(k)$.*

For a fixed 0–n-path P in H, Lemma 8 yields $\mathbb{E}[C_\alpha(\pi_P)] = \sum_{v \in V} \mathbb{E}[C_{\alpha,v}(\pi_P)] \leq \sum_{k=0}^{n} \psi_P(k)$. Recall that in Step 3 of the concatenating algorithm, we pick the traversal direction of each subtour such that $\sum_{v \in V_j} C_{\alpha,v}(\pi_{\text{ALG}} + \pi_{\min})$ is minimized, where $V_j \coloneqq V_{n_j} \setminus \left(\bigcup_{i=1}^{j-1} V_{n_i}\right)$. Hence, for the tour π_{ALG} we obtain the deterministic bound

$$C_\alpha(\pi_{\text{ALG}}) \leq \sum_{v \in V} C_{\alpha,v}(\pi_{P^*}) \leq \sum_{k=0}^{n} \psi_{P^*}(k). \tag{2}$$

Next, we use the following identity, which expresses the length of a path P in H in terms of ψ_P. Its proof can be found in the full version of the paper.

▶ **Lemma 9.** *For a 0–n-path P in H we have $\sum_{k=0}^{n} \psi_P(k) = \ell(P)$.*

The following lemma is the technically most challenging part of the analysis, as it compares the length of a 0–n-path P in H against the total α-latency of the optimal solution. The proof uses similar ideas as in [15] and [22]. We consider a randomized sequence of good k-trees with exponentially increasing costs where $\gamma \in (1, \infty)$ is a factor determining the rate of the exponential growth. Then the expected length of that path can be bounded by a constant factor times the optimal total α-latency. A probabilistic argument then yields that this bound is also attained by the shortest 0–n-path P^* in H. Specifically, we obtain the following bound. Its proof can be found in the full version of the paper.

▶ **Lemma 10.** *Let $\pi^* \in \Pi$ be an optimal solution for the discounted graph search problem. Then there exists a 0–n-path P in H such that $\ell(P) \leq 2\frac{\gamma+\alpha}{(1+\alpha)\ln\gamma} C_\alpha(\pi^*)$ for any $\gamma \in (1, \infty)$.*

Lemma 10 yields an approximation ratio parametrized in $\gamma \in (1, \infty)$. To obtain the smallest approximation ratio possible, we optimize over γ. To this end, we denote by $W : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ the Lambert-W function that assigns $x \geq 0$ the unique value $w \geq 0$ such that $we^w = x$. The proof of the following result can be found in the full version of the paper.

▶ **Lemma 11.** *For $\gamma \in (1, \infty)$, let h be defined as $h(\gamma) \coloneqq 2\frac{\gamma+\alpha}{(1+\alpha)\ln(\gamma)}$. Then h is minimized for $\gamma^* = e^{1+W(\alpha/e)}$. In particular, we have $\varrho_1(\alpha) = h(\gamma^*)$.*

Combining all previous results yields the proof for Lemma 7; this proof can be found in the full version of the paper.

We have shown so far that the concatenating algorithm with input $\mathcal{T}'$ (the set of trees guaranteed by Lemma 6 with adjusted costs for phantom trees) computes a solution to the discounted graph search problem with approximation guarantee $\varrho_1(\alpha)$. However, we obtained this result only under the assumption that all trees were real trees, i.e., that they could be transformed into tours. It remains to show that we can obtain the same approximation guarantee when using only real trees as input to the concatenating algorithm. The issue with phantom trees as input arises only if phantom trees lie on the shortest 0–n-path P^* computed by the concatenating algorithm, since these trees are the ones traversed to obtain the final tour π_{ALG}. As a first step towards proving Theorem 2, we thus show that whenever the shortest 0–n-path P^* on input $\mathcal{T}'$ contains a phantom tree, there exists another 0–n-path with the same length that visits only real trees. The proof of the lemma follows the argument of [4, 15] and can be found in the full version of the paper.

▶ **Lemma 12.** *Let H be the auxiliary graph constructed by the concatenating algorithm on input $\mathcal{T}'$. Then there exists a shortest 0–n-path P in H that does not visit any vertex whose corresponding tree is a phantom tree.*

By Lemma 12, applying the concatenating algorithm to the set $\mathcal{R} \subseteq \mathcal{T}$ of real trees in $\mathcal{T}$ ensures that the algorithm is well-defined while maintaining an approximation guarantee of $\varrho_1(\alpha)$. We can now prove Theorem 2; the proof can be found in the full version of the paper.

4 Algorithms for the p-Norm of α-Latencies

We now turn to the case of arbitrary $p \geq 1$. The case $p = 1$ was treated in the previous section, where the linearity of the objective allowed us to use an auxiliary graph whose edge lengths directly represent the additional latency caused by adding a tree. For $p > 1$, this linearity is lost, and the same construction no longer applies directly.

At a high level, we would still like to use sequences of trees of geometrically increasing cost. However, two difficulties arise. First, as pointed out by Garg [19], it is not possible, for every $k \in [n]$, to use the primal–dual algorithm to obtain a k-MST with cost no more than the optimum k-stroll. In particular, this issue occurs for those values of k for which the primal–dual procedure does not directly compute a k-MST. Second, when $p > 1$, the objective is nonlinear in the vertices' α-latencies. Thus, an analogue of the auxiliary-graph cost from the previous section has to keep track not only of how many vertices have already been visited, but also of the accumulated cost of the trees used so far.

We subdivide this section into two parts. First, we present a randomized algorithm based on a geometric scaling argument. Then, we show how to derandomize it using a time-expanded auxiliary graph; the resulting running time is pseudo-polynomial in the edge costs.

4.1 A Randomized Algorithm

To avoid the difficulties with the auxiliary-graph construction for the moment, we first give a randomized algorithm with the desired approximation guarantee.

▶ **Theorem 3.** *For every $p \geq 1$, there is a polynomial-time randomized algorithm for the discounted graph search problem with approximation factor*

$$\varrho_p(\alpha) = \min_{\gamma > 1} \left\{ \frac{2(1+\alpha)}{\gamma - 1} \left[\frac{\gamma^{2p} - \gamma^p}{p \ln \gamma} \right]^{1/p} \right\}.$$

The approximation ratio given in the above theorem is depicted in Section 1.1. It ranges from 2e for $\alpha = 0$ and $p = 1$ to a limiting value of 16 for $\alpha = 1$ as $p \to \infty$. Note that the value 2e matches the bound for $\alpha = 0$ and $p = 1$ from the previous section.

Instead of computing a *good* k-tree, we use Garg's 2-approximation [19] to compute a 2-approximate k-MST for every $k \in [n]$. Recall that such a tree contains s and visits k additional vertices. Let these trees be denoted by $T_1, T_2, \ldots, T_n$.

We next define a randomized sequence of trees, similar to the construction in the previous section. Fix some $\gamma > 1$, which will be optimized later. Let $b = \gamma^U$, where U is drawn uniformly at random from $[0, 1]$. For all relevant values of r, define $n_r = \max\{k \in [n]_0 : c(T_k) \leq 2b\gamma^r\}$. Here, $c(T_k)$ denotes only the total length of the tree and, in particular, does not depend on p.

Let $P = (n_0, n_1, \ldots, n_l)$ be the sequence of values obtained by the above procedure, where $n_0 = 0$. As in the previous section, we concatenate the sequence of trees induced by P to obtain a tour, which we denote by π_P. We also define a function ψ_P such that $\psi_P(k)$ gives an upper bound on the α-latency of the k-th distinct vertex in $V \setminus \{s\}$ visited by π_P. Formally, the function $\psi_P : [n]_0 \to \mathbb{R}_{\geq 0}$ is defined by

$$\psi_P(k) \coloneqq \begin{cases} 0 & \text{if } k = 0, \\ (1+\alpha) \sum_{i=0}^{j} c(T_{n_i}) & \text{if } n_{j-1} < k \leq n_j, \text{for some } j \in \{1, \ldots, l\}. \end{cases}$$

We obtain the following lemma.

▶ **Lemma 13.** *Let π_P be the randomized tour obtained from P, and let $v \in V$ be the k-th distinct vertex of $V \setminus \{s\}$ in π_P, for some $k \in [n]_0$. Then $C_{\alpha,v}(\pi_P) \leq \psi_P(k)$.*

Proof. This follows directly from the definition of ψ_P. In the worst case, all trees are nested and vertex k is visited at the very end of the last tree containing it. Moreover, the cost of the tour induced by tree T_{n_i} is bounded by $(1+\alpha)c(T_{n_i})$. ◀

Lemma 13 and the randomized sequence of trees imply Theorem 3.

Proof of Theorem 3. Let π^* be an optimal tour for the p-norm α-latency objective. Let v_i^* be the i-th vertex visited by π^*. Then

$$\mathrm{OPT} = \left(\sum_{i \in [n]} \left(C_{\alpha, v_i^*}(\pi^*) \right)^p \right)^{1/p}.$$

Fix some arbitrary $i \in [n]$. Let r be such that $C_{\alpha, v_i^*}(\pi^*) = d\gamma^r$ for some $d \in [1, \gamma)$. We consider two cases.

First case: $d \le b$. In this case, there is a tree T_{n_r} with cost at most $2b\gamma^r$ and $n_r \ge i$. Hence,

$$\psi_P(i) \le (1+\alpha) \sum_{k=1}^{r} c(T_{n_k}) \le (1+\alpha) \sum_{k=1}^{r} 2b\gamma^k < (1+\alpha) \sum_{k=-\infty}^{r} 2b\gamma^k = (1+\alpha) 2b\gamma^r \frac{\gamma}{\gamma - 1}.$$

Second case: $d > b$. In this case, the same argument applied to the next threshold gives

$$\psi_P(i) \le (1+\alpha) \sum_{k=1}^{r+1} c(T_{n_k}) \le (1+\alpha) \sum_{k=1}^{r+1} 2b\gamma^k < (1+\alpha) \sum_{k=-\infty}^{r+1} 2b\gamma^k = (1+\alpha) 2b\gamma^{r+1} \frac{\gamma}{\gamma - 1}.$$

Since U is chosen uniformly at random in $[0, 1]$ and $b = \gamma^U$, we can average over the two cases and obtain

$$\mathbb{E}_U[\psi_P(i)^p] \le \int_{\log_\gamma d}^{1} \left((1+\alpha) 2b\gamma^r \frac{\gamma}{\gamma - 1} \right)^p \mathrm{d}U + \int_{0}^{\log_\gamma d} \left((1+\alpha) 2b\gamma^{r+1} \frac{\gamma}{\gamma - 1} \right)^p \mathrm{d}U.$$

We now compute the right-hand side. Let $a \coloneqq \log_\gamma d$. Since $b = \gamma^U$, we have

$$\begin{aligned}
&\int_{\log_\gamma d}^{1} \left((1+\alpha) 2b\gamma^r \frac{\gamma}{\gamma - 1} \right)^p \mathrm{d}U + \int_{0}^{\log_\gamma d} \left((1+\alpha) 2b\gamma^{r+1} \frac{\gamma}{\gamma - 1} \right)^p \mathrm{d}U \\
&= \left((1+\alpha) 2 \frac{\gamma}{\gamma - 1} \right)^p \left[\gamma^{rp} \int_a^1 \gamma^{pU} \, \mathrm{d}U + \gamma^{(r+1)p} \int_0^a \gamma^{pU} \, \mathrm{d}U \right].
\end{aligned}$$

Using

$$\int \gamma^{pU} \, \mathrm{d}U = \frac{\gamma^{pU}}{p \ln \gamma},$$

we obtain

$$\begin{aligned}
&\left((1+\alpha) 2 \frac{\gamma}{\gamma - 1} \right)^p \left[\gamma^{rp} \int_a^1 \gamma^{pU} \, \mathrm{d}U + \gamma^{(r+1)p} \int_0^a \gamma^{pU} \, \mathrm{d}U \right] \\
&= \left((1+\alpha) 2 \frac{\gamma}{\gamma - 1} \right)^p \frac{\gamma^{rp}(\gamma^p - \gamma^{pa}) + \gamma^{(r+1)p}(\gamma^{pa} - 1)}{p \ln \gamma}.
\end{aligned}$$

Since $a = \log_\gamma d$, we have $\gamma^{pa} = d^p$. Hence,

$$\begin{aligned}
\mathbb{E}_U[\psi_P(i)^p] &\le \left((1+\alpha) 2 \frac{\gamma}{\gamma - 1} \right)^p \frac{\gamma^{rp}(\gamma^p - d^p) + \gamma^{(r+1)p}(d^p - 1)}{p \ln \gamma} \\
&= \left((1+\alpha) 2 \frac{\gamma}{\gamma - 1} \right)^p \frac{d^p \gamma^{rp}(\gamma^p - 1)}{p \ln \gamma} \\
&= (1+\alpha)^p \frac{2^p}{p \ln \gamma} \left[\left(\frac{d\gamma^{r+2}}{\gamma - 1} \right)^p - \left(\frac{d\gamma^{r+1}}{\gamma - 1} \right)^p \right].
\end{aligned}$$

The contribution of v_i^* to the optimum is

$$\left(C_{\alpha,v_i^*}(\pi^*)\right)^p = (d\gamma^r)^p.$$

Therefore, the previous bound implies

$$\frac{\mathbb{E}_U[\psi_P(i)^p]}{(d\gamma^r)^p} \leq (1+\alpha)^p \frac{2^p}{p\ln\gamma}\left[\left(\frac{\gamma^2}{\gamma-1}\right)^p - \left(\frac{\gamma}{\gamma-1}\right)^p\right] = (1+\alpha)^p \frac{2^p}{p\ln\gamma}\cdot\frac{\gamma^{2p}-\gamma^p}{(\gamma-1)^p} =: \eta. \tag{3}$$

In particular, since $C_{\alpha,v_i^*}(\pi^*) = d\gamma^r$, we have

$$\mathbb{E}_U[\psi_P(i)^p] \leq \eta \cdot \left(C_{\alpha,v_i^*}(\pi^*)\right)^p. \tag{4}$$

We now use this bound to compare the expected p-norm α-latency of the randomized tour with the optimum. By the definition of ψ_P, for every realization of U we have

$$C_{\alpha,v_i^*}(\pi_P) \leq \psi_P(i) \qquad \text{for all } i \in [n].$$

Hence,

$$\mathbb{E}_U\left[\left(\sum_{i\in[n]} C_{\alpha,v_i^*}(\pi_P)^p\right)^{1/p}\right] \leq \mathbb{E}_U\left[\left(\sum_{i\in[n]} \psi_P(i)^p\right)^{1/p}\right].$$

We next use Jensen's inequality. Recall that if f is concave and X is a nonnegative random variable, then

$$\mathbb{E}[f(X)] \leq f(\mathbb{E}[X]).$$

In our setting, we apply this with

$$f(x) = x^{1/p} \qquad \text{and} \qquad X = \sum_{i\in[n]} \psi_P(i)^p.$$

Since $p \geq 1$, the function $f(x) = x^{1/p}$ is concave on $\mathbb{R}_{\geq 0}$. Moreover, $X \geq 0$ for every realization of the random choice of U. Therefore, Jensen's inequality gives

$$\mathbb{E}_U\left[\left(\sum_{i\in[n]} \psi_P(i)^p\right)^{1/p}\right] \leq \left(\mathbb{E}_U\left[\sum_{i\in[n]} \psi_P(i)^p\right]\right)^{1/p} = \left(\sum_{i\in[n]} \mathbb{E}_U[\psi_P(i)^p]\right)^{1/p}.$$

Using (4), we obtain

$$\begin{aligned}
\mathbb{E}_U\left[\left(\sum_{i\in[n]} C_{\alpha,v_i^*}(\pi_P)^p\right)^{1/p}\right] &\leq \left(\sum_{i\in[n]} \eta\cdot\left(C_{\alpha,v_i^*}(\pi^*)\right)^p\right)^{1/p} \\
&= \eta^{1/p}\left(\sum_{i\in[n]} \left(C_{\alpha,v_i^*}(\pi^*)\right)^p\right)^{1/p} \\
&= \eta^{1/p}\cdot \mathrm{OPT}.
\end{aligned}$$

It remains to simplify $\eta^{1/p}$. By (3),

$$\begin{aligned}\eta^{1/p} &= (1+\alpha)\left[\frac{2^p}{p\ln\gamma}\cdot\frac{\gamma^{2p}-\gamma^p}{(\gamma-1)^p}\right]^{1/p}\\ &= (1+\alpha)\frac{2}{(p\ln\gamma)^{1/p}}\cdot\frac{(\gamma^{2p}-\gamma^p)^{1/p}}{\gamma-1}\\ &= (1+\alpha)\frac{2}{\gamma-1}\left[\frac{\gamma^{2p}-\gamma^p}{p\ln\gamma}\right]^{1/p}.\end{aligned}$$

Thus, for every fixed $\gamma > 1$, the randomized algorithm has expected approximation factor

$$(1+\alpha)\frac{2}{\gamma-1}\left[\frac{\gamma^{2p}-\gamma^p}{p\ln\gamma}\right]^{1/p}.$$

Optimizing over $\gamma > 1$ gives the claimed bound. ◀

4.2 Derandomization via a Time-Expanded Auxiliary Graph

We now describe how to derandomize the randomized algorithm from the previous subsection. The idea is similar to the derandomization used for the case $p = 1$: we construct an auxiliary graph and compare the length of a shortest path in this graph to the expected length of a suitable randomized path.

For $p = 1$, the auxiliary graph only needs to keep track of how many vertices have already been visited. For $p > 1$, however, the objective is nonlinear in the latencies. Thus, the additional cost of traversing a tree depends not only on the number of vertices visited so far, but also on the latency accumulated so far. We therefore use a two-dimensional auxiliary graph. One dimension keeps track of the number of vertices that have already been visited, and the other dimension keeps track of the total cost of the trees traversed so far.

▶ **Theorem 4.** *For every $p \geq 1$, there is a deterministic pseudo-polynomial-time algorithm for the discounted graph search problem with approximation factor $\varrho_p(\alpha)$ defined in Theorem 3. Moreover, if the tree costs are integral and polynomially bounded, then the algorithm runs in polynomial time.*

Let $T_1, \ldots, T_n$ be the 2-approximate k-MST trees used in the previous subsection, and let $c(T_k)$ denote the cost of tree T_k. We set T_0 to be the trivial tree containing only s, with $c(T_0) = 0$. Let

$$Z \coloneqq \left\{\sum_{k\in S} c(T_k) : S \subseteq [n]\right\}.$$

Thus, Z is the set of all values that can occur as the accumulated cost of a set of traversed trees.

We construct a directed auxiliary graph $H = (V_H, A_H)$ as follows: The vertex set is $V_H = \big([n]_0 \times Z\big) \cup \{\omega\}$, where ω is an additional terminal vertex. A vertex (i, z) represents the state in which i vertices of $V \setminus \{s\}$ have already been visited, and the total cost of the trees traversed so far is z. For every $i, j \in [n]_0$ with $i < j$ and every $z, z' \in Z$ with $z' = z + c(T_j)$, we add the directed edge $\big((i, z), (j, z')\big)$. The length of this edge is defined as

$$\ell_{(i,z),(j,z')} \coloneqq (n-i)\left(\big((1+\alpha)z'\big)^p - \big((1+\alpha)z\big)^p\right). \tag{5}$$

Finally, for every $z \in Z$, we add an edge $((n, z), \omega)$ of length zero. The algorithm computes a shortest path from $(0, 0)$ to ω in H. Let $P^* = \big((n_0, z_0), (n_1, z_1), \ldots, (n_l, z_l), \omega\big)$ be such a shortest path, where $n_0 = 0$, $z_0 = 0$, and $n_l = n$. The algorithm then concatenates the trees $T_{n_1}, T_{n_2}, \ldots, T_{n_l}$ in this order, using the tree-to-tour conversion from Proposition 5. We denote the resulting tour by π_{ALG}. We first relate the length of a path in H to the p-norm α-latency of the corresponding concatenated tour.

▶ **Lemma 14.** *Let $P = \big((n_0, z_0), (n_1, z_1), \ldots, (n_l, z_l), \omega\big)$ be any $(0, 0)$–ω path in H, where $n_0 = 0$, $z_0 = 0$, and $n_l = n$. Let π_P be the tour obtained by concatenating the trees $T_{n_1}, \ldots, T_{n_l}$. Then $\big(\sum_{v \in V} C_{\alpha,v}(\pi_P)^p\big)^{1/p} \leq \ell(P)^{1/p}$.*

Proof. For $h \in \{1, \ldots, l\}$, the transition from (n_{h-1}, z_{h-1}) to (n_h, z_h) corresponds to traversing tree T_{n_h}, and by construction

$$z_h = z_{h-1} + c(T_{n_h}).$$

After this traversal, the accumulated cost of the traversed trees is z_h. By the tree-to-tour conversion, the α-latency of every vertex that is first covered by T_{n_h} is at most

$$(1 + \alpha) z_h.$$

Hence, if $n_{h-1} < k \leq n_h$, then the k-th distinct vertex visited by π_P has α-latency at most $(1 + \alpha) z_h$.

Define

$$\psi_P(k) \coloneqq (1 + \alpha) z_h \qquad \text{if } n_{h-1} < k \leq n_h.$$

Then

$$C_{\alpha, v_k}(\pi_P) \leq \psi_P(k)$$

for the k-th distinct vertex v_k of $V \setminus \{s\}$ visited by π_P. Therefore,

$$\sum_{v \in V} C_{\alpha,v}(\pi_P)^p \leq \sum_{k=1}^{n} \psi_P(k)^p.$$

It remains to observe that the right-hand side is exactly the length of P. Indeed, using the definition of the edge lengths in (5), we get

$$\begin{aligned}
\ell(P) &= \sum_{h=1}^{l} (n - n_{h-1}) \left(\big((1 + \alpha) z_h\big)^p - \big((1 + \alpha) z_{h-1}\big)^p \right) \\
&= \sum_{h=1}^{l} (n_h - n_{h-1}) \big((1 + \alpha) z_h\big)^p \\
&= \sum_{k=1}^{n} \psi_P(k)^p.
\end{aligned}$$

The second equality follows by telescoping. Thus,

$$\sum_{v \in V} C_{\alpha,v}(\pi_P)^p \leq \ell(P),$$

which proves the claim. ◀

We next show that the auxiliary graph contains a path whose length is bounded by the same expression that appeared in the randomized analysis. For fixed $\gamma > 1$, define

$$\eta_\gamma \coloneqq (1+\alpha)^p \frac{2^p}{p \ln \gamma} \cdot \frac{\gamma^{2p} - \gamma^p}{(\gamma - 1)^p}.$$

▶ **Lemma 15.** *Let π^* be an optimal tour for the p-norm α-latency objective. Then, for every fixed $\gamma > 1$, there exists a $(0,0)$–ω path P in H such that*

$$\ell(P) \le \eta_\gamma \sum_{i \in [n]} \big(C_{\alpha, v_i^*}(\pi^*)\big)^p.$$

Proof. We use the randomized construction from the previous subsection. For a random choice of $U \in [0,1]$, let $P(U)$ be the corresponding sequence of trees. After removing consecutive repetitions, this sequence defines a path in the auxiliary graph H. Indeed, the first coordinate records the number of vertices covered, while the second coordinate records the accumulated cost of the selected trees.

By the definition of the auxiliary graph and the calculation in the proof of Lemma 14, we have

$$\ell(P(U)) = \sum_{i \in [n]} \psi_{P(U)}(i)^p.$$

From the randomized analysis, for every $i \in [n]$ we have

$$\mathbb{E}_U[\psi_{P(U)}(i)^p] \le \eta_\gamma \big(C_{\alpha, v_i^*}(\pi^*)\big)^p.$$

Taking the sum over all $i \in [n]$ and using linearity of expectation gives

$$\mathbb{E}_U[\ell(P(U))] = \mathbb{E}_U\left[\sum_{i \in [n]} \psi_{P(U)}(i)^p\right] = \sum_{i \in [n]} \mathbb{E}_U[\psi_{P(U)}(i)^p] \le \eta_\gamma \sum_{i \in [n]} \big(C_{\alpha, v_i^*}(\pi^*)\big)^p.$$

Therefore, there exists at least one realization of U such that

$$\ell(P(U)) \le \eta_\gamma \sum_{i \in [n]} \big(C_{\alpha, v_i^*}(\pi^*)\big)^p.$$

This proves the lemma. ◀

Using the above lemmas and analyzing the size of the time-expanded auxiliary graph, we prove Theorem 4.

Proof of Theorem 4. Let P^* be a shortest $(0,0)$–ω path in H, and let π_{ALG} be the tour obtained by concatenating the trees along P^*. By Lemma 14,

$$\left(\sum_{v \in V} C_{\alpha, v}(\pi_{\mathrm{ALG}})^p\right)^{1/p} \le \ell(P^*)^{1/p}.$$

Since P^* is a shortest path, for every $\gamma > 1$ and for the path P whose existence is guaranteed by Lemma 15, we have

$$\ell(P^*) \le \ell(P).$$

Thus,

$$\ell(P^*) \le \eta_\gamma \sum_{i\in[n]} \left(C_{\alpha,v_i^*}(\pi^*)\right)^p = \eta_\gamma \cdot \mathrm{OPT}^p.$$

Taking p-th roots gives

$$\left(\sum_{v\in V} C_{\alpha,v}(\pi_{\mathrm{ALG}})^p\right)^{1/p} \le \eta_\gamma^{1/p} \cdot \mathrm{OPT}.$$

Finally,

$$\eta_\gamma^{1/p} = (1+\alpha)\frac{2}{\gamma-1}\left[\frac{\gamma^{2p}-\gamma^p}{p\ln\gamma}\right]^{1/p}.$$

Since this holds for every $\gamma > 1$, we may minimize over $\gamma > 1$, which gives the claimed approximation factor.

It remains to discuss the running time. Let

$$N_H := |V_H| + |A_H|$$

denote the size of the auxiliary graph. All edge lengths in H are nonnegative, and hence a shortest $(0,0)$–ω path can be computed with Dijkstra's algorithm in time

$$O(N_H \log N_H).$$

In general, the set Z may be exponentially large, since it consists of all subset sums of the tree costs. Thus, the algorithm is pseudo-polynomial. If the tree costs are integral and polynomially bounded, then $|Z|$ is polynomially bounded as well. In this case, N_H is polynomially bounded, and the algorithm runs in polynomial time. ◀

5 Discussion

We studied the discounted graph search problem, a two-parameter framework that contains pathwise search, expanding search, TSP-type objectives, and MST-type objectives as special or limiting cases. The first parameter, $\alpha \in [0,1]$, determines how repeated traversals of edges are charged, while the second parameter, $p \ge 1$, determines how the individual vertex latencies are aggregated. For $p = 1$, we obtain a polynomial-time approximation algorithm whose guarantee interpolates between the currently best-known approximation ratios for expanding search and pathwise search. For general $p \ge 1$, we obtain a randomized constant-factor approximation and a deterministic pseudo-polynomial-time algorithm with the same approximation guarantee.

We stated and analyzed the problem for a uniform discount factor α. The results for $p = 1$ also extend to the case where every edge e has its own discount factor $\alpha_e \in [0,1]$. In this setting, the first traversal of edge e costs c_e, while every further traversal costs $\alpha_e c_e$. Let $\alpha_{\min} := \min_{e\in E} \alpha_e$ and $\alpha_{\max} := \max_{e\in E} \alpha_e$. Then the analysis yields the approximation guarantee

$$\varrho_1(\alpha) := \begin{cases} 2\mathrm{e} & \text{if } \alpha_{\max} = \alpha_{\min} = 0, \\ 2\dfrac{\alpha_{\max}}{(1+\alpha_{\min})W(\alpha_{\max}/\mathrm{e})} & \text{otherwise.} \end{cases}$$

The reason is that the analysis uses the largest possible discount factor when comparing the cost of trees to paths, while the conversion of trees into tours benefits from the smallest discount factor. Thus, the same proof goes through with α replaced by $\alpha_{\max}$ in the former part of the analysis and by $\alpha_{\min}$ in the latter.

A similar generalization is possible when the cost of an edge depends on how often it has already been used. Suppose that for each edge e we are given a sequence $c_e^{(1)}, c_e^{(2)}, c_e^{(3)}, \ldots$ such that the k-th traversal of edge e costs $c_e^{(k)}$. If these sequences are non-increasing, i.e., $c_e^{(k)} \geq c_e^{(k+1)}$ for all $e \in E$ and $k \in \mathbb{N}$, then our analysis can again be extended. Define $\alpha_{\min} := \min_{e \in E} c_e^{(2)}/c_e^{(1)}$ and $\alpha_{\max} := \max_{e \in E} c_e^{(2)}/c_e^{(1)}$. Then the same approximation guarantee as above follows. Indeed, in the comparison to the optimum, it is sufficient to consider paths in which each edge is used at most twice. On the other hand, the algorithm may use edges more often, and for non-increasing traversal costs, every traversal after the second one can only become cheaper. Thus, replacing all costs $c_e^{(k)}$ for $k \geq 2$ by $c_e^{(2)}$ can only make the algorithm more expensive, while preserving the relevant comparison to the optimum.

By contrast, for arbitrary non-decreasing traversal-cost sequences, one cannot hope for a constant-factor approximation in full generality. For example, suppose that $c_e^{(1)} = 0$ and $c_e^{(k)} = 1$ for all $k \geq 2$ for every edge $e \in E$. Then there is a traversal with total latency zero if and only if the graph has a Hamiltonian path starting at s. Otherwise, every feasible traversal must repeat some edge and hence has positive latency. Since deciding the existence of such a Hamiltonian path is NP-complete, no constant-factor approximation can exist in this setting unless P = NP. It would be interesting to understand which structural assumptions on the sequences $c_e^{(1)}, c_e^{(2)}, c_e^{(3)}, \ldots$ still allow constant-factor approximations.

Finally, for the case $p > 1$, one can also compare our bounds to algorithms for the limiting case $p = \infty$. For any nonnegative latency vector $x \in \mathbb{R}^n_{\geq 0}$, we have $\|x\|_\infty \leq \|x\|_p \leq n^{1/p}\|x\|_\infty$. Thus, any β-approximation for the maximum-latency objective immediately gives a $\beta n^{1/p}$-approximation for the p-norm objective. At the endpoint $\alpha = 1$, the maximum-latency objective corresponds to a TSP-path-type problem, while at the endpoint $\alpha = 0$ it corresponds to the MST problem. Therefore, for large values of p, one can use approximation algorithms for these limiting problems to obtain alternative approximation guarantees. In particular, once $p = \Omega(\log n)$, the factor $n^{1/p}$ becomes constant. For even larger values of p, this factor approaches 1, and the guarantee approaches that of the corresponding maximum-latency approximation algorithm. This gives better bounds for sufficiently large p in regimes where the limiting TSP- or MST-type objective can be approximated more accurately than the general bound obtained from the randomized geometric construction.

References

1 Foto N. Afrati, Stavros S. Cosmadakis, Christos H. Papadimitriou, George Papageorgiou, and Nadia Papakostantinou. The complexity of the travelling repairman problem. *RAIRO – Theoretical Informatics and Applications*, 20(1):79–87, 1986. doi:10.1051/ita/1986200100791.

2 Steve Alpern and Thomas Lidbetter. Mining coal or finding terrorists: The expanding search paradigm. *Operations Research*, 61(2):265–279, 2013. doi:10.1287/opre.1120.1134.

3 Aaron Archer and Anna Blasiak. Improved approximation algorithms for the minimum latency problem via prize-collecting strolls. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 429–447, 2010. doi:10.1137/1.9781611973075.36.

4 Aaron Archer, Asaf Levin, and David P. Williamson. A faster, better approximation algorithm for the minimum latency problem. *SIAM Journal on Computing*, 37(5):1472–1498, 2008. doi:10.1137/07068151X.

5 Sanjeev Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998. doi:10.1145/290179.290180.

6 Sanjeev Arora and George Karakostas. Approximation schemes for minimum latency problems. *SIAM Journal on Computing*, 32(5):1317–1337, 2003. doi:10.1137/S009753970139965.

7 Sanjeev Arora and George Karakostas. A $2+\varepsilon$ approximation algorithm for the k-MST problem. *Mathematical Programming*, 107(3):491–504, 2006. doi:10.1007/S10107-005-0693-1.

8 Sunil Arya and H. Ramesh. A 2.5-factor approximation algorithm for the k-MST problem. *Information Processing Letters*, 65:117–118, 1998. doi:10.1016/S0020-0190(98)00010-6.

9 Igor Averbakh and Jordi Pereira. The flowtime network construction problem. *IIE Transactions*, 44(8):681–694, 2012. doi:10.1080/0740817X.2011.636792.

10 Baruch Awerbuch, Yossi Azar, Edward F Grove, Ming-Yang Kao, P Krishnan, and Jeffrey Scott Vitter. Load balancing in the L_p norm. In *Proceedings of IEEE 36th Symposium on Foundations of Computer Science*, pages 383–391. IEEE, 1995. doi:10.1109/SFCS.1995.492494.

11 Yossi Azar, Leah Epstein, Yossi Richter, and Gerhard J Woeginger. All-norm approximation algorithms. *Journal of Algorithms*, 52(2):120–133, 2004. doi:10.1016/j.jalgor.2004.02.003.

12 Avrim Blum, Prasad Chalasani, Don Coppersmith, William R. Pulleyblank, Prabhakar Raghavan, and Madhu Sudan. The minimum latency problem. In *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC)*, pages 163–171, 1994. doi:10.1145/195058.195125.

13 Deeparnab Chakrabarty and Chaitanya Swamy. Interpolating between k-median and k-center: Approximation algorithms for ordered k-median. In *Proceedings of the 45th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 29:1–29:14, 2018. doi:10.4230/LIPIcs.ICALP.2018.29.

14 Deeparnab Chakrabarty and Chaitanya Swamy. Approximation algorithms for minimum norm and ordered optimization problems. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing*, pages 126–137, 2019. doi:10.1145/3313276.3316322.

15 Kamalika Chaudhuri, Brighten Godfrey, Satish Rao, and Kunal Talwar. Paths, trees, and minimum latency tours. In *Proceedings of the Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 36–45, 2003. doi:10.1109/SFCS.2003.1238179.

16 Majid Farhadi, Alejandro Toriello, and Prasad Tetali. The traveling firefighter problem. In *Proceedings of the SIAM Conference on Applied and Computational Discrete Algorithms (ACDA)*, pages 205–216. SIAM, 2021. doi:10.1137/1.9781611976830.19.

17 Alfredo García, Pedro Jodrá, and Javier Tejel. A note on the travelling repairman problem. *Networks*, 40:27–31, 2002. doi:10.1002/net.10031.

18 Naveen Garg. A 3-approximation for the minimum tree spanning k vertices. In *Proceedings of the Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 302–309, 1996. doi:10.1109/SFCS.1996.548489.

19 Naveen Garg. Saving an epsilon: A 2-approximation for the k-MST problem in graphs. In *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC)*, pages 396–402, 2005. doi:10.1145/1060590.1060650.

20 Michel X. Goemans and Jon M. Kleinberg. An improved approximation ratio for the minimum latency problem. *Mathematical Programming*, 82:111–124, 1998. doi:10.1007/BF01585867.

21 Daniel Golovin, Anupam Gupta, Amit Kumar, and Kanat Tangwongsan. All-norms and all-L_p-norms approximation algorithms. In *Proceedings of the 28th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 199–210, 2008. doi:10.4230/LIPIcs.FSTTCS.2008.1753.

22 Svenja M. Griesbach, Felix Hommelsheim, Max Klimm, and Kevin Schewior. Improved approximation algorithms for the expanding search problem. *SIAM Journal on Discrete Mathematics*, 40(1):349–373, 2026. doi:10.1137/25M1759902.

23 Ben Hermans, Roel Leus, and Jannik Matuschke. Exact and approximation algorithms for the expanding search problem. *INFORMS Journal on Computing*, 34(1):281–296, 2022. doi:10.1287/ijoc.2020.1047.

24 Sharat Ibrahimpur and Chaitanya Swamy. Minimum-norm load balancing is (almost) as easy as minimizing makespan. In *Proceedings of the 48th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 81–1, 2021. doi:10.4230/LIPIcs.ICALP.2021.81.

25 Edward Minieka. The delivery man problem on a tree network. *Annals of Operations Research*, 18:261–266, 1989. doi:10.1007/BF02097807.

26 Sartaj Sahni and Teofilo Gonzalez. P-complete approximation problems. *Journal of the ACM*, 23:555–565, 1976. doi:10.1145/321958.321975.

27 René Sitters. The minimum latency problem is NP-hard for weighted trees. In *Proceedings of the International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 230–239, 2002. doi:10.1007/3-540-47867-1_17.

28 René Sitters. Polynomial time approximation schemes for the traveling repairman and other minimum latency problems. *SIAM Journal on Computing*, 50(5):1580–1602, 2021. doi:10.1137/19M126918X.

Threshold Rounding and Bounded-Degree Boolean MAX 2-CSP

Suprovat Ghoshal
Indiana University, Bloomington, IN, USA

Neng Huang
University of Michigan, Ann Arbor, MI, USA

Euiwoong Lee
University of Michigan, Ann Arbor, MI, USA

Konstantin Makarychev
Northwestern University, Evanston, IL, USA

Yury Makarychev
Toyota Technological Institute at Chicago, IL, USA

Abstract

We describe an $\widetilde{\Omega}(1/d^4)$-improvement over threshold rounding schemes for a broad class of Boolean MAX 2-CSP instances in which every variable appears in at most d constraints. In the case of MAX 2-SAT, we improve the ratio further and obtain an $(\beta_\star + \widetilde{\Omega}(1/d^2))$-factor approximation algorithm for bounded-degree MAX 2-SAT instances, where $\beta_\star$ is the UGC-optimal approximation ratio for MAX 2-SAT achieved by the LLZ algorithm [20]. Our result generalizes an $(\alpha_{GW} + \widetilde{\Omega}(1/d^2))$-factor approximation algorithm for MAX CUT on graphs with degrees bounded by d, due to Hsieh and Kothari [15]. Together with the state-of-the-art approximability results for MAX DI-CUT and MAX 2-AND [7], our result suggests that similar improvements exist for bounded-degree instances of these problems as well.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis

Keywords and phrases MAX 2-SAT, Approximation Algorithms, Constraint Satisfaction Problems

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.28

Category APPROX

Related Version *Full Version*: `https://arxiv.org/abs/2607.11050`

Funding *Euiwoong Lee*: Supported in part by NSF award CCF-2236669.
Konstantin Makarychev: Supported by NSF awards CCF-1955351 and EECS-2216970.
Yury Makarychev: Supported by NSF award EECS-2216899

1 Introduction

Maximum Constraint Satisfaction Problems (MAX CSPs) are a fundamental class of combinatorial problems that are studied extensively in theoretical computer science. The design of approximation algorithms for MAX CSPs has been a central topic of research that has seen substantial progress in the past decades. Some of the main highlights include the design of approximation algorithms for MAX CSPs [21] and the development of a framework for hardness reductions to study the limits of approximation for CSPs [17, 18]. These developments have culminated in surprisingly tight characterizations of the optimal approximation factor of every MAX CSP [23], under the Unique Games Conjecture (UGC) [16].

In this work, we aim to explore the approximability of MAX CSPs for structurally simpler instances, namely, when the instances are *sparse*. Specifically, we explore the question of whether one can improve on the optimal approximation guarantee of a MAX CSP when the degree of the underlying constraint graph is bounded. Formally, we are interested in the following:


© Suprovat Ghoshal, Neng Huang, Euiwoong Lee, Konstantin Makarychev, and Yury Makarychev;
licensed under Creative Commons License CC-BY 4.0



Given a MAX CSP problem P, suppose its optimal approximation factor is α_P. Then, if the degree of the instance is bounded by some constant d, can we efficiently achieve $\alpha_P + \epsilon_d$ approximation, where the improvement ϵ_d depends on the degree d?

The above question has been studied for various specific MAX CSPs in the literature [6, 13, 9, 4, 5, 15]. Arguably, one of the most interesting lines of work among these involves the approximability of MAX CUT in this setting. It was first studied by Feige, Karpinski, and Langberg [9], who showed that the MAX CUT problem on graphs with maximum degree d admits an efficient approximation guarantee of $\alpha_{GW} + \Omega(1/d^4)$, where $\alpha_{GW} \approx 0.87856$ denotes the optimal approximation factor for MAX CUT under the UGC [12, 19]. This was then improved by Florén [11] up to an $\alpha_{GW} + \Omega(1/d^3)$ factor, and by Hsieh and Kothari [15] to a factor of $\alpha_{GW} + \widetilde{\Omega}(1/d^2)$. Another notable work in this context is that of [5], who showed that one can bypass the approximation resistance of MAX k-LIN and other related problems in bounded-degree instances.

Unfortunately, generalizing this phenomenon beyond the specific examples mentioned above has been challenging. In particular, such results are not known even for well-studied friends of the MAX CUT problem, such as MAX 2-SAT and MAX 2-AND. One of the main difficulties in obtaining similar guarantees for general MAX CSPs has been the following: most of the aforementioned improvements in the bounded-degree setting implicitly rely on tight structural characterizations of the hardest-to-round, i.e., integrality gap, instances of the problem[1]. Such explicit characterizations are known only for a handful of examples (e.g., MAX CUT [10], MAX k-LIN [14], MAX 2-SAT [20, 8]), and even the question of tightly characterizing the gap instances for every Boolean 2-CSP is a long-standing open question.

To get around this bottleneck, we instead consider improving the approximation guarantee of a family of algorithms in the degree-bounded setting. Specifically, we consider the family of $\mathcal{THRESH}^-$-rounding scheme-based algorithms described informally below[2]:

SDP + $\mathcal{THRESH}^-$-Rounding Algorithm

1. Solve the canonical semi-definite programming (SDP) relaxation for the MAX CSP.
2. Round the SDP solution using the $\mathcal{THRESH}^-$ family of rounding functions.

Figure 1 The $\mathcal{THRESH}^-$-based Approximation Framework.

The above framework was first introduced by Lewin, Livnat, and Zwick [20], and it vastly generalizes the hyperplane rounding-based algorithm of Goemans and Williamson [12]. This algorithmic framework has been used successfully for obtaining optimal and near-optimal guarantees for a wide variety of MAX CSPs such as MAX 2-SAT [20, 8], MAX 2-AND [7], MAX BISECTION [3], and several other related problems. Furthermore, Austrin showed that under a certain positivity condition, the above framework is optimal for approximating Boolean MAX 2-CSPs assuming UGC [2]. Guided by these considerations, we are therefore motivated to ask – *Can we improve on the approximation guarantee of the $\mathcal{THRESH}^-$-based algorithm, when the degree of the input instance is bounded?* – which is the main question explored by this work.

[1] We elaborate the connection between integrality gaps and improvements in the bounded-degree setting in Section 1.2.

[2] We refer the reader to Section 2 for a more formal description of the SDP relaxation and the $\mathcal{THRESH}^-$ rounding scheme.

1.1 Our Results

In this work, we make progress on the above questions and give improved approximation guarantees for bounded-degree Boolean MAX 2-CSP instances which satisfy a natural "reasonable" condition. The formal definition of the reasonable condition is a bit technical, and we instead attempt to give an informal description of it. Let $\mathcal{P}$ denote a set of Boolean predicates, and let MAX CSP($\mathcal{P}$) denote the MAX CSP problem where the predicates corresponding to the constraints belong to $\mathcal{P}$. Then loosely speaking, we say that the MAX CSP($\mathcal{P}$) problem is reasonable if for any hardest-to-round instance[3] of MAX CSP($\mathcal{P}$), the local geometric arrangement of the vector solution in the neighborhood of any vertex satisfies certain "general position" conditions. In particular, these conditions ensure that certain pairwise moments involving labels of variables in the neighborhood are non-constant under the rounding distribution (see Lemma 22). We refer the reader to Definitions 6 and 20 for a formal description of this condition.

We now state our main result, which gives improved approximation guarantees for reasonable CSPs.

▶ **Theorem 1** (Informal version of Theorem 23). *Suppose $\mathcal{P}$ is a set of Boolean binary predicates such that the MAX CSP($\mathcal{P}$) problem is reasonable. Let $\beta_{\mathcal{P}}$ denote the approximation factor achieved by the algorithm in Figure 1 for MAX CSP($\mathcal{P}$). Let I be an instance of MAX CSP($\mathcal{P}$) such that every variable appears in at most d constraints. Then, there exists an algorithm that achieves an approximation factor of $\beta_{\mathcal{P}} + \widetilde{\Omega}(d^{-4})$ on I, where the constant hidden in $\widetilde{\Omega}$ only depends on $\mathcal{P}$.*

While the above theorem by itself just implies improvement on the approximation guarantee of $\mathcal{THRESH}^{-}$-based algorithms, it has stronger implications for MAX CSPs that satisfy certain additional properties. As mentioned above, Austrin [2] showed that if $\mathcal{P}$ satisfies the positivity condition, then the algorithm from Figure 1 obtains the optimal approximation guarantee assuming UGC. Therefore, for MAX CSP($\mathcal{P}$) problems that satisfy the reasonable and positivity conditions, the above theorem actually implies improvement on the optimal approximation guarantee in the bounded-degree setting.

Furthermore, by leveraging the structural insights from the recent work [8], we are able to derive a sharper version of the above theorem for the MAX 2-SAT problem. We formally state this result in the next theorem.

▶ **Theorem 2.** *Let I be an instance of MAX 2-SAT such that every variable appears in at most d constraints. Then there is an efficient algorithm that can return a $\beta_\star + \Omega(d^{-2}\log^{-1}(d))$-approximation to I, where $\beta_\star \approx 0.9401$ is the optimal approximation guarantee for the MAX 2-SAT problem.*

We point out a couple of interesting features of the above theorem. Firstly, observe that the above theorem does not mention the "reasonable" condition. This is because the structural results from [8] directly imply that the hardest-to-round instances of MAX 2-SAT naturally satisfy the reasonable condition, and hence we are able to apply our framework to obtain an unconditional improvement for degree-bounded MAX 2-SAT instances. Furthermore, by using the specific structure of the 2-SAT predicate, we are able to improve the additive gain in the approximation factor from $\widetilde{\Omega}(d^{-4})$ to $\widetilde{\Omega}(d^{-2})$.

[3] An instance I of MAX CSP($\mathcal{P}$) is said to be hardest-to-round if the algorithm from Figure 1 achieves its worst approximation guarantee in I.

1.2 On Integrality Gaps and Bounded-degree CSPs

We close out this section by highlighting the connection between hard-to-round instances and improved approximation for bounded-degree CSPs. Most of the previous works in this setting adopt the approach of improving the performance of a base algorithm via a greedy post-processing step that relies on certain anti-concentration properties of the objective function at local neighborhoods (similar to Lemma 18). In particular, such an approach has to be capable of interpolating between the following extremal scenarios:

- For instances that are far from being an integrality gap, the base algorithm by itself is capable of yielding improved approximation guarantees,
- And for instances that are close to being an integrality gap, the post-processing procedure should be able to leverage the bounded-degree structure of the instance to improve on the worst-case approximation guarantee.

Now, note that under this framework, it stands to reason that the greedy post-processing needs to be more effective for instances that are closer to being an integrality gap, whereas it may be less useful for easier instances – e.g., for instances which are perfectly solvable by the base algorithm, the procedure cannot improve the quality of the solution output by the base algorithm any further. Therefore, the analysis of the post-processing step necessarily requires a structural understanding of the hardest instances for the base algorithm. For example, for the MAX CUT problem, the analyses of algorithms in [9, 11, 15] use the well-known characterization that the edges that are hardest to round by hyperplane rounding are those for which the inner product of the corresponding SDP vectors is equal to a certain critical value.

We conclude by noting that while it might be possible to obtain improvements without explicitly characterizing the hardest instances, such a result will require algorithms that go beyond the above two-phase approach.

1.3 Organization of the Paper

The rest of the paper is organized as follows. In Section 2, we formally define MAX CSP, the Basic SDP relaxation, the $\mathcal{THRESH}^-$ rounding family, and related concepts. In Section 3, we present our algorithm and define some algorithmic quantities. In Section 4, we analyze the algorithm in the case of MAX 2-SAT, proving Theorem 2. In Section 5, we analyze our algorithm for general Boolean MAX 2-CSPs, where we state and prove the formal version of Theorem 1. Finally, we state in Appendix A certain variance lower bounds needed in the analysis, whose proofs we defer to the full version of the paper.

2 Preliminaries

For $n \in \mathbb{N}$, let $[n] \coloneqq \{1, 2, \ldots, n\}$. Let Φ, φ be the c.d.f. and p.d.f. of a standard Gaussian variable. Given an event A, we use $\mathbf{1}\{A\}$ to denote the indicator random variable for A. We use `true`, `false` for the truth values of a Boolean variable. We may also represent the truth values using $\{0, 1\}$ or $\{-1, 1\}$: in the former case 1 stands for `true` and 0 stands for `false`, while in the latter case -1 stands for `true` and 1 stands for `false`.

2.1 MAX CSP and the Basic SDP Relaxation

▶ **Definition 3.** *Let $\mathcal{P}$ be a set of Boolean predicates. An instance $I = (V, \mathcal{C}, w)$ of* MAX CSP($\mathcal{P}$) *is defined by*

- *A finite set V of variables.*

- *A finite set $\mathcal{C}$ of constraints where each constraint $C \in \mathcal{C}$ is obtained by applying some predicate P to some variables from V. We use $C = (P, i_1, i_2, \ldots, i_k)$ to denote a constraint C obtained by applying some k-ary predicate P to variables $i_1, i_2, \ldots, i_k$.*
- *A weight function $w : \mathcal{C} \to \mathbb{R}^{\geq 0}$.*

Given an assignment $A : V \to \{-1, 1\}$, we say that a constraint C is satisfied if the corresponding predicate evaluates to 1, and we use $\mathrm{Val}(I, A)$ to denote the total weight of constraints satisfied by A. The objective is to find an assignment A that maximizes $\mathrm{Val}(I, A)$.

For simplicity, we will assume without loss of generality that $\sum_{C \in \mathcal{C}} w(C) = 1$. This can always be achieved by renormalizing, which does not meaningfully change the problem. In this case we may also think of $\mathcal{C}$ as a distribution over constraints, with probabilities specified by w. We will also assume $V = [n]$ for some $n \in \mathbb{N}$.

In this paper we only consider the special case of MAX 2-CSP, namely the case where every Boolean predicate in $\mathcal{P}$ is of arity 2. For convenience, we encode truth assignments for variables using $\{-1, 1\}$, while predicates take values in $\{0, 1\}$, and thus each predicate has the form $P : \{-1, 1\}^2 \to \{0, 1\}$. Given a MAX 2-CSP instance $I = (V, \mathcal{C}, w)$, we can formulate its *Basic SDP relaxation* [23] as follows (cf. [8]).

$$
\begin{aligned}
\text{maximize} \quad & \sum_{C=(P,i,j)\in\mathcal{C}} w(C) \cdot \left(\hat{P}_\varnothing + \hat{P}_i \mathbf{v}_0 \cdot \mathbf{v}_i + \hat{P}_j \mathbf{v}_0 \cdot \mathbf{v}_j + \hat{P}_{i,j} \mathbf{v}_i \cdot \mathbf{v}_j \right) \\
\text{subject to} \quad & \forall i \in \{0, 1, 2, \ldots, n\} \,, \qquad \mathbf{v}_i \cdot \mathbf{v}_i = 1 \,, \\
& \forall C = (P, i, j) \in \mathcal{C} \,, \qquad \begin{aligned} (\mathbf{v}_0 - \mathbf{v}_i) \cdot (\mathbf{v}_0 - \mathbf{v}_j) &\geq 0 \,, \\ (\mathbf{v}_0 + \mathbf{v}_i) \cdot (\mathbf{v}_0 - \mathbf{v}_j) &\geq 0 \,, \\ (\mathbf{v}_0 - \mathbf{v}_i) \cdot (\mathbf{v}_0 + \mathbf{v}_j) &\geq 0 \,, \\ (\mathbf{v}_0 + \mathbf{v}_i) \cdot (\mathbf{v}_0 + \mathbf{v}_j) &\geq 0 \,. \end{aligned}
\end{aligned}
$$

Here $\hat{P}_\varnothing, \hat{P}_i, \hat{P}_j, \hat{P}_{i,j} \in \mathbb{R}$ are the *Fourier coefficients* of P, namely, we have

$$P(x_i, x_j) = \hat{P}_\varnothing + \hat{P}_i x_i + \hat{P}_j x_j + \hat{P}_{i,j} x_i x_j.$$

for every $x_i, x_j \in \{-1, 1\}$ (See [22] for more on Fourier analysis of Boolean functions). The constraints in (2.1) are often referred to as *triangle inequalities*. In fact, we may in addition impose triangle inequalities on all triples of vectors:

$$
\forall i, j, k \in \{0, 1, \ldots, n\} \,, \qquad \begin{aligned} (\mathbf{v}_i - \mathbf{v}_j) \cdot (\mathbf{v}_i - \mathbf{v}_k) &\geq 0 \,, \\ (\mathbf{v}_i + \mathbf{v}_j) \cdot (\mathbf{v}_i - \mathbf{v}_k) &\geq 0 \,, \\ (\mathbf{v}_i - \mathbf{v}_j) \cdot (\mathbf{v}_i + \mathbf{v}_k) &\geq 0 \,, \\ (\mathbf{v}_i + \mathbf{v}_j) \cdot (\mathbf{v}_i + \mathbf{v}_k) &\geq 0 \,. \end{aligned}
$$

▶ **Definition 4.** *We say that unit vectors $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n$ are an SDP solution for some MAX 2-CSP instance over the variable set $[n]$, if they satisfy (2.1). We say that they satisfy all triangle inequalities if they further satisfy (2.1). For $i, j \in [n]$, we write*

$$b_i = \mathbf{v}_0 \cdot \mathbf{v}_i \qquad \text{and} \qquad b_{ij} = \mathbf{v}_i \cdot \mathbf{v}_j.$$

We will often need to take projections for an SDP solution. For $i \in [n]$ and $S \subseteq \{0\} \cup [n]$, we use $\mathbf{v}_i^{\perp\{S\}}$ to denote the (renormalized) component in $\mathbf{v}_i$ that is orthogonal to $\mathrm{span}\{\mathbf{v}_j \mid j \in S\}$. In particular, $\mathbf{v}_i^{\perp\{0\}} = \frac{\mathbf{v}_i - b_i \mathbf{v}_0}{\sqrt{1-b_i^2}}$ is the (renormalized) component in $\mathbf{v}_i$ that is orthogonal to $\mathbf{v}_0$ (in the case $|b_i| = 1$, we may define $\mathbf{v}_i^{\perp\{0\}}$ to be any unit vector that is orthogonal to all other vectors in the SDP solution, but this case won't show up in our paper). We will write $\mathbf{v}_i^\perp$ for $\mathbf{v}_i^{\perp\{0\}}$ to avoid clutter.

2.2 Configurations

To analyze SDP-based approximation algorithms, it is sometimes more convenient to work with the inner products between vectors in an SDP solution rather than the vectors themselves. This leads to the following definition.

▶ **Definition 5.** *A* configuration *is a tuple of the form (b_i, b_j, b_{ij}, P) where P is a Boolean predicate of arity 2, and $b_i, b_j, b_{ij} \in [-1, 1]$ are real numbers such that*

$$-1 + |b_i + b_j| \leq b_{ij} \leq 1 - |b_i - b_j|.$$

We say that a tuple (b_i, b_j, b_{ij}) is valid *if it satisfies (5). Given a configuration $\theta = (b_i, b_j, b_{ij}, P)$, we define $\rho(\theta) = \frac{b_{ij} - b_i b_j}{\sqrt{(1-b_i^2)(1-b_j^2)}}$ if $\sqrt{(1-b_i^2)(1-b_j^2)} \neq 0$ and $\rho(\theta) = 0$ otherwise.*

It can be easily verified that the inequality (5) corresponds exactly to the triangle inequalities in (2.1), and therefore a tuple (b_i, b_j, b_{ij}) is valid if and only if it can be obtained as inner products between $\mathbf{v}_0, \mathbf{v}_i, \mathbf{v}_j$ from an SDP solution. In this case, $\rho(\theta)$ is exactly the inner product between $\mathbf{v}_i^\perp$ and $\mathbf{v}_j^\perp$.

▶ **Definition 6.** *Given a MAX 2-CSP instance $I = (V, \mathcal{C}, w)$ and an SDP solution $\mathbf{v}_0, \cdots, \mathbf{v}_n$, we say that a distribution $\mathcal{D}$ of configurations* represents *the SDP solution if it is obtained as follows:*

- *Sample a constraint $C = (P, i, j) \sim \mathcal{C}$ from the instance.*
- *Return the configuration $\theta = (b_i, b_j, b_{ij}, P)$.*

We may also say that θ as defined above represents the constraint C.

Now we define *completeness*, which corresponds to the value of the Basic SDP relaxation.

▶ **Definition 7** (Completeness)**.** *Given a configuration $\theta = (b_i, b_j, b_{ij}, P)$, we define its* completeness *to be*

$$\mathsf{Completeness}(\theta) = \hat{P}_\varnothing + \hat{P}_i b_i + \hat{P}_j b_j + \hat{P}_{i,j} b_{ij}.$$

For a distribution $\mathcal{D}$ over configurations, we define $\mathsf{Completeness}(\mathcal{D}) = \mathbb{E}_{\theta \sim \mathcal{D}}[\mathsf{Completeness}(\theta)]$.

By construction, if a distribution $\mathcal{D}$ represents some SDP solution, then its completeness corresponds exactly to the SDP objective value achieved by that SDP solution.

2.3 The $\mathcal{THRESH}^-$ Rounding Family

The $\mathcal{THRESH}^-$ rounding family, introduced in [20], is a powerful family of rounding algorithms for Boolean MAX 2-CSPs. A rounding algorithm from the $\mathcal{THRESH}^-$ rounding family is specified by a *threshold function* $f : [-1, 1] \to \mathbb{R}$. Given an SDP solution $\mathbf{v}_0, \mathbf{v}_1, \ldots, \mathbf{v}_n$, it computes the projection $\mathbf{v}_i^\perp = \frac{\mathbf{v}_i - b_i \mathbf{v}_0}{\sqrt{1-b_i^2}}$ for every $i \in [n]$, samples a standard Gaussian vector $\mathbf{r}$, and sets the variable i to be `true` if $\mathbf{r} \cdot \mathbf{v}_i^\perp \geq f(b_i)$ and `false` otherwise.

We now define *soundness*, which corresponds to the probability that a configuration (or more precisely, any constraint represented by the configuration) is satisfied using a $\mathcal{THRESH}^-$ rounding algorithm.

▶ **Definition 8** (Soundness). *Let $f : [-1, 1] \to \mathbb{R}$ be a threshold function. Given a configuration $\theta = (b_i, b_j, b_{ij}, P)$, let (g_i, g_j) be a pair of standard Gaussian random variables with correlation $\rho(\theta)$, and define*

$$X_i = \begin{cases} -1, & \text{if } g_i \geq f(b_i), \\ 1, & \text{otherwise,} \end{cases} \qquad X_j = \begin{cases} -1, & \text{if } g_j \geq f(b_j), \\ 1, & \text{otherwise,} \end{cases}$$

Then, the soundness *of θ with respect to f is defined to be*

$$\mathsf{Soundness}_f(\theta) = \Pr[P(X_i, X_j) = \texttt{true}].$$

For a distribution $\mathcal{D}$ over configurations, we define $\mathsf{Soundness}_f(\mathcal{D}) = \mathbb{E}_{\theta \sim \mathcal{D}}[\mathsf{Soundness}_f(\theta)]$.

By construction, if a distribution $\mathcal{D}$ represents some SDP solution, then its soundness with respect to f corresponds to the expected satisfied fraction if we use $\mathcal{THRESH}^-$ with f to round that SDP solution.

3 Description of the Algorithm

We now state our main algorithm. It is adapted from the bounded-degree MAX CUT algorithm due to [15]. The quantities $W_{i,1}, W_{i,0}$, and $W_{i,*}$ will be defined momentarily after the algorithm.

Algorithm 1 Algorithm for Bounded-Degree Boolean MAX 2-CSP.

1: **Input:** Boolean MAX 2-CSP instance $I = (V, \mathcal{C}, w)$ with variable set $V = [n]$, parameter $\epsilon > 0$, and an SDP solution $\{\mathbf{v}_i \mid i \in V \cup \{0\}\}$.
2: **Output:** an assignment $A : V \to \{\texttt{true}, \texttt{false}\}$.
3: Choose a standard Gaussian vector $\mathbf{r}$.
4: **for** $i \in V$ **do**
5: Compute $b_i = \mathbf{v}_0 \cdot \mathbf{v}_i$ and $\mathbf{v}_i^\perp = \frac{\mathbf{v}_i - b_i \mathbf{v}_0}{\sqrt{1-b_i^2}}$.
6: Set $A'(i) = \texttt{true}$ if $\mathbf{r} \cdot \mathbf{v}_i^\perp \geq f(b_i)$ and $\texttt{false}$ otherwise.
7: Set $A(i) = A'(i)$ if $\left|\mathbf{r} \cdot \mathbf{v}_i^\perp - f(b_i)\right| \geq \epsilon$ and $*$ otherwise.
8: **for** $i \in V$ **do**
9: **if** $A(i) = *$ **then**
10: Set $A(i) = \begin{cases} \texttt{true} & \text{if } W_{i,1} > W_{i,0} + W_{i,*}, \\ \texttt{false} & \text{if } W_{i,0} > W_{i,1} + W_{i,*}, \\ A'(i) & \text{otherwise.} \end{cases}$
11: **return** A

Informally, the algorithm first computes an assignment A' using $\mathcal{THRESH}^-$ with the threshold function f. It then greedily flips any variable which lies approximately on the boundary of the $\mathcal{THRESH}^-$ rounding algorithm (more specifically, $\left|\mathbf{r} \cdot \mathbf{v}_i^\perp - f(b_i)\right| < \epsilon$) if there is some *local gain*, which we define below.

For every variable i, let $\mathcal{C}_i \subseteq \mathcal{C}$ be the set of constraints that involve i. We define the following subsets of $\mathcal{C}_i$ with respect to the (partial) assignment A before executing the for-loop at Line 8:

$$\begin{aligned}
\mathcal{C}_{i,*} &= \{C \in \mathcal{C}_i \mid \text{the other variable in } C \text{ is unassigned}\} \\
\mathcal{C}_{i,\mathtt{sat}} &= \{C \in \mathcal{C}_i \setminus \mathcal{C}_{i,*} \mid C \text{ is satisfied regardless of the truth value of } i\} \\
\mathcal{C}_{i,\mathtt{unsat}} &= \{C \in \mathcal{C}_i \setminus \mathcal{C}_{i,*} \mid C \text{ is unsatisfied regardless of the truth value of } i\} \\
\mathcal{C}_{i,1} &= \{C \in \mathcal{C}_i \setminus \mathcal{C}_{i,*} \mid C \text{ is satisfied if } i \text{ is } \mathtt{true} \text{ and unsatisfied if } i \text{ is } \mathtt{false}\} \\
\mathcal{C}_{i,0} &= \{C \in \mathcal{C}_i \setminus \mathcal{C}_{i,*} \mid C \text{ is satisfied if } i \text{ is } \mathtt{false} \text{ and unsatisfied if } i \text{ is } \mathtt{true}\}.
\end{aligned}$$

We use $W_{i,*}, W_{i,\mathtt{sat}}, W_{i,\mathtt{unsat}}, W_{i,1}, W_{i,0}$ to denote the total weights of clauses in these subsets, and we define

$$W_i \coloneqq \sum_{C \in \mathcal{C}_i} w(C) = W_{i,*} + W_{i,\mathtt{sat}} + W_{i,\mathtt{unsat}} + W_{i,1} + W_{i,0}$$

to be the total weight of constraints which contain i. We would like to understand how the for-loop starting at Line 8 affects the constraints in these subsets. By definition, constraints in $\mathcal{C}_{i,\mathtt{sat}}$ and $\mathcal{C}_{i,\mathtt{unsat}}$ won't be affected by the truth value for i since they are already fixed, and therefore they (as well as the weights $W_{i,\mathtt{sat}}, W_{i,\mathtt{unsat}}$) won't play a role in our analysis. The only constraints whose status can be affected in the for-loop are the ones in $\mathcal{C}_{i,*}, \mathcal{C}_{i,0}$ and $\mathcal{C}_{i,1}$. If we set i to be $\mathtt{true}$, then we satisfy every constraint in $\mathcal{C}_{i,1}$, possibly some constraints in $\mathcal{C}_{i,*}$, and certainly none in $\mathcal{C}_{i,0}$. Similarly, if we set i to be $\mathtt{false}$, then we satisfy every constraint in $\mathcal{C}_{i,0}$, possibly some constraints in $\mathcal{C}_{i,*}$, and certainly none in $\mathcal{C}_{i,1}$. It follows that if we flip i from $\mathtt{true}$ to $\mathtt{false}$, then the total weight of satisfied constraints changes by $W_{i,0} - W_{i,1} - p \cdot W_{i,*}$, for some $p \in [-1, 1]$. If we have $W_{i,0} - W_{i,1} - W_{i,*} > 0$, then this change is guaranteed to be positive. In particular, this means when $W_{i,0} - W_{i,1} - W_{i,*} > 0$ and $A'(i) = \mathtt{true}$, we obtain an improvement over A' by setting $A(i) = \mathtt{false}$. Analogously, when $W_{i,1} - W_{i,0} - W_{i,*} > 0$ and $A'(i) = \mathtt{false}$, we also obtain an improvement over A' by setting $A(i) = \mathtt{true}$. This motivates the formal definition of *local gains* as follows:

▶ **Definition 9.** *We define the local gain Δ_i of the variable i to be*

$$\begin{aligned}
\Delta_i = (W_{i,0} - W_{i,1} - W_{i,*})_+ \cdot \mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i), f(b_i) + \epsilon)\} \\
+ (W_{i,1} - W_{i,0} - W_{i,*})_+ \cdot \mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i) - \epsilon, f(b_i))\}.
\end{aligned}$$

Here $(z)_+$ is defined to be z if $z \geq 0$ and 0 if $z < 0$.

Note that there is tension over the choice of ϵ, which controls how many vertices we potentially flip in the for-loop: on one hand we would like to flip more vertices in order to get more local gain, on the other hand we would like $W_{i,*}$ to be small, for otherwise it may overwhelm the difference between $W_{i,0}$ and $W_{i,1}$ and we won't get any local gain at all.

The following lemma formalizes the motivation for local gains.

▶ **Lemma 10.** *Let $\mathcal{D}$ be the distribution of configurations that represents the SDP solution in the input to Algorithm 1. Then, we have $\mathbb{E}[\mathrm{Val}(I, A)] \geq \mathsf{Soundness}_f(\mathcal{D}) + \sum_{i \in V} \mathbb{E}[\Delta_i]$.*

Proof. Observe that for any $i \in V$, we have $A'(i) = A(i)$ unless

- $W_{i,1} > W_{i,0} + W_{i,*}$ and $\mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i) - \epsilon, f(b_i))$, or
- $W_{i,0} > W_{i,1} + W_{i,*}$ and $\mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i), f(b_i) + \epsilon)$.

In these two cases, we flip the value of variable i and the weight of satisfied constraints increases by at least Δ_i. So we have

$$\mathrm{Val}(I, A) - \mathrm{Val}(I, A') \geq \sum_{i \in V} \Delta_i.$$

The lemma follows by taking expectation and observing that $\mathbb{E}[\mathrm{Val}(I, A')] = \mathsf{Soundness}_f(\mathcal{D})$. ◀

4 Analysis for MAX 2-SAT

We first analyze Algorithm 1 in the case of MAX 2-SAT. Let $\mathcal{P}_{2SAT} = \{x \vee y, \overline{x} \vee y, \overline{x} \vee \overline{y}\}$, then MAX 2-SAT is exactly MAX CSP($\mathcal{P}_{2SAT}$). We start by stating some known results for MAX 2-SAT. For $\beta \in [0,1]$, define the threshold function $f_\beta : [-1,1] \to \mathbb{R}$ by

$$f_\beta(t) = \Phi^{-1}\left(\frac{1+\beta t}{2}\right),$$

where Φ^{-1} is the inverse CDF for a standard Gaussian variable. Observe that f_β is odd for any $\beta \in [0,1]$, in the sense that $f_\beta(-t) = -f_\beta(t)$ for every $t \in [-1,1]$.

▶ **Theorem 11** ([20, 1, 8]). *There exists some constant $\beta_\star \approx 0.94016567$ such that for any MAX 2-SAT configuration θ,*

$$\mathsf{Soundness}_{f_{\beta_\star}}(\theta) - \beta_* \cdot \mathsf{Completeness}(\theta) \geq 0.$$

Consequently, the $\mathcal{THRESH}^-$ algorithm with the $f = f_{\beta_\star}$ achieves a $\beta_\star$-approximation for MAX 2-SAT. On the other hand, assuming the Unique Games Conjecture, for any $\epsilon > 0$ it is NP-hard to approximate MAX 2-SAT within a factor of $\beta_\star + \epsilon$.

By inspecting the proofs in [8] we in fact have the following slightly stronger statement:

▶ **Theorem 12** ([8]). *Given a valid triple (b_i, b_j, b_{ij}), define*

$$g(b_i, b_j, b_{ij}) = \mathsf{Soundness}_{f_{\beta_\star}}(\theta) - \beta_* \cdot \mathsf{Completeness}(\theta),$$

where $\theta = (b_i, b_j, b_{ij}, x \vee y)$. Then g has exactly two minimizers

$$g(b_\star, b_\star, -1+2b_\star) = g(-b_\star, -b_\star, -1+2b_\star) = 0$$

where $b_\star \approx 0.162478$. Furthermore, there exists $c > 0$ such that for any valid tuple (b_i, b_j, b_{ij}),

$$\begin{aligned} &g(b_i, b_j, b_{ij}) \\ &\geq c \cdot \left(\min\left(\|(b_i, b_j, b_{ij}) - (b_\star, b_\star, -1+2b_\star)\|, \|(b_i, b_j, b_{ij}) - (-b_\star, -b_\star, -1+2b_\star)\|\right)\right)^2. \end{aligned}$$

Let $\theta = (b_i, b_j, b_{ij}, P)$ be a MAX 2-SAT configuration (i.e., $P \in \mathcal{P}_{2SAT}$) and $\delta > 0$. We say that θ is δ-critical, if one of the following holds:

- $P = x \vee y$ or $P = \overline{x} \vee \overline{y}$, and

$$\min\left(\|(b_i, b_j, b_{ij}) - (b_\star, b_\star, -1+2b_\star)\|, \|(b_i, b_j, b_{ij}) - (-b_\star, -b_\star, -1+2b_\star)\|\right) \leq \delta.$$

- $P = \overline{x} \vee y$ and

$$\min\left(\|(b_i, b_j, b_{ij}) - (b_\star, -b_\star, 1-2b_\star)\|, \|(b_i, b_j, b_{ij}) - (-b_\star, b_\star, 1-2b_\star)\|\right) \leq \delta.$$

Theorem 12 shows that for non-critical configurations we may hope for a better-than-$\beta_\star$ approximation.

▶ **Lemma 13.** *Let θ be a MAX 2-SAT configuration θ that is not δ-critical. Then*

$$\mathsf{Soundness}_{f_{\beta_\star}}(\theta) - \beta_\star \cdot \mathsf{Completeness}(\theta) = \Omega(\delta^2).$$

Proof. Follows directly from Theorem 12. ◀

On the other hand, for δ-critical configurations with sufficiently small $\delta > 0$, we may use Algorithm 1 to obtain an improvement if the instance is bounded degree.

▶ **Theorem 14.** *There exists some constant $\delta > 0$ such that the following holds. Let $I = (V, \mathcal{C}, w)$ be an instance of MAX 2-SAT such that every variable appears in at most d constraints. Let $\mathcal{D}$ be a distribution of configurations representing some SDP solution $\{\mathbf{v}_i \mid i \in V \cup \{0\}\}$ for I such that $b_i \geq 0$ for every variable i, and every $\theta \in \mathrm{supp}(\mathcal{D})$ is δ-critical. Then, for $f = f_{\beta_\star}$ and $\epsilon = d^{-1}\log^{-1} d$, Algorithm 1 returns an assignment A such that $\mathbb{E}[\mathrm{Val}(I, A)] = \mathsf{Soundness}_f(\mathcal{D}) + \Omega(d^{-2}\log^{-1}(d))$.*

We remark that the assumption that $b_i \geq 0$ for every variable i is for convenience only. The proof of Theorem 14 is deferred to Section 4.1. Combining Lemma 13 and Theorem 14 we obtain the following theorem:

▶ **Theorem 15** (Restatement of Theorem 2)**.** *Let I be an instance of MAX 2-SAT such that every variable appears in at most d constraints. Let $\mathcal{D}$ be a distribution of configurations representing an SDP solution for I. Then, for $f = f_{\beta_\star}$, we can find an assignment A such that*

$$\mathrm{Val}(I, A) \geq \beta_\star \cdot \mathsf{Completeness}(\mathcal{D}) + \Omega(d^{-2}\log^{-1}(d)).$$

Proof. Let $\delta > 0$ be chosen as in Theorem 14 and $\eta = cd^{-2}\log^{-1}(d)$ for some small constant $c > 0$. If there are at least η-fraction (in the weighted sense) of configurations in $\mathcal{D}$ that are not δ-critical, then in this case by Lemma 13 we have that the $\mathcal{THRESH}^-$ algorithm with the $f = f_{\beta_\star}$ already achieves $\mathsf{Soundness}_f(\mathcal{D}) \geq \beta_\star \cdot \mathsf{Completeness}(\mathcal{D}) + \Omega(\eta) = \beta_\star \cdot \mathsf{Completeness}(\mathcal{D}) + \Omega(d^{-2}\log^{-1}(d))$.

Now assume at least $(1-\eta)$-fraction of the configurations are δ-critical, then we may ignore the constraints corresponding to configurations that are not δ-critical and form a new instance I' and a new distribution of configurations $\mathcal{D}'$ supported only on δ-critical configurations by renormalizing the weights. Applying Theorem 14 to I' and $\mathcal{D}'$ (after possibly flipping variables and SDP vectors so that $b_i \geq 0$ for every i), we obtain an assignment achieving a satisfying fraction of at least

$$\begin{aligned}
&(1-\eta)\cdot\big(\mathsf{Soundness}_f(\mathcal{D}') + \Omega(d^{-2}\log^{-1}(d))\big)\\
\geq\ &(1-\eta)\cdot\big(\beta_\star\cdot\mathsf{Completeness}(\mathcal{D}') + \Omega(d^{-2}\log^{-1}(d))\big)\\
\geq\ &(1-\eta)\cdot\big(\beta_\star\cdot(\mathsf{Completeness}(\mathcal{D}) - \eta) + \Omega(d^{-2}\log^{-1}(d))\big)\\
=\ &\beta_\star\cdot\mathsf{Completeness}(\mathcal{D}) + \Omega(d^{-2}\log^{-1}(d)) - \eta\cdot O(1)\\
=\ &\beta_\star\cdot\mathsf{Completeness}(\mathcal{D}) + \Omega(d^{-2}\log^{-1}(d)).
\end{aligned}$$

◀

4.1 Analyzing a Local Neighborhood

In this subsection we aim to prove Theorem 14. Since we assume $b_i \geq 0$ for every variable i, our δ-critical configurations (b_i, b_j, b_{ij}, P) must all satisfy that $\|(b_i, b_j, b_{ij}) - (b_\star, b_\star, -1+2b_\star)\| \leq \delta$ and P is either $x \vee y$ or $\overline{x} \vee \overline{y}$, if we take δ to be sufficiently small.

To prove Theorem 14, we first state and prove a few lemmas on the algorithmic quantities $W_{i,0}, W_{i,1}, W_{i,*}, \Delta_i$ in the neighborhood of i defined in Section 3. To avoid ambiguity, we use A_* to denote the (partial) assignment A before the execution of the for-loop at Line 8. The overall idea is to show that conditioned on i being unassigned, $W_{i,0} - W_{i,1}$ must have a large variance, leading to a nontrivial local gain.

▶ **Lemma 16.** *For every i and $t \in (f(b_i) - \epsilon, f(b_i) + \epsilon)$, we have*

$$\mathbb{E}[|W_{i,0} - W_{i,1}| \mid \mathbf{r} \cdot \mathbf{v}_i^{\perp} = t] = W_i \cdot \Omega(d^{-1}),$$

where the constant in Ω can be chosen independently from t.

Proof. Let $\mathcal{C}_i^+ \subseteq \mathcal{C}_i$ be the set of constraints involving i that are defined using the predicate $x \vee y$, and $\mathcal{C}_i^- \subseteq \mathcal{C}_i$ be the set of those defined using $\overline{x} \vee \overline{y}$. For $C \in \mathcal{C}_i$, let j_C be the variable other than i involved in C. For every $C \in \mathcal{C}_i^+$, we have

$$C \in \begin{cases} \mathcal{C}_{i,\mathtt{sat}} & \text{if } A_*(j_C) = \mathtt{true}, \\ \mathcal{C}_{i,1} & \text{if } A_*(j_C) = \mathtt{false}, \\ \mathcal{C}_{i,*} & \text{if } A_*(j_C) = *. \end{cases}$$

Similarly, for every $C \in \mathcal{C}_i^-$, we have

$$C \in \begin{cases} \mathcal{C}_{i,\mathtt{sat}} & \text{if } A_*(j_C) = \mathtt{false}, \\ \mathcal{C}_{i,0} & \text{if } A_*(j_C) = \mathtt{true}, \\ \mathcal{C}_{i,*} & \text{if } A_*(j_C) = *. \end{cases}$$

So we have

$$W_{i,0} - W_{i,1} = \sum_{C \in \mathcal{C}_i^+} w(C)\mathbf{1}\{A_*(j_C) = \mathtt{true}\} - \sum_{C \in \mathcal{C}_i^-} w(C)\mathbf{1}\{A_*(j_C) = \mathtt{false}\}.$$

Given a configuration $\theta = (b_i, b_j, b_{ij}, P)$, we have $A_*(j) = \mathtt{true}$ if and only if $\mathbf{r} \cdot \mathbf{v}_j^{\perp} \geq f(b_j) + \epsilon$, or equivalently $\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}} \geq \frac{-\rho(\theta)\mathbf{r}\cdot\mathbf{v}_i^{\perp} + f(b_j) + \epsilon}{\sqrt{1-\rho(\theta)^2}}$. Similarly $A_*(j) = \mathtt{false}$ if and only if $\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}} \leq \frac{-\rho(\theta)\mathbf{r}\cdot\mathbf{v}_i^{\perp} + f(b_j) - \epsilon}{\sqrt{1-\rho(\theta)^2}}$. Since $\mathbf{r} \cdot \mathbf{v}_i^{\perp} \in (f(b_i) - \epsilon, f(b_i) + \epsilon)$ and θ is δ-critical, we have $\| \frac{-\rho(\theta)\mathbf{r}\cdot\mathbf{v}_i^{\perp} + f(b_j) \pm \epsilon}{\sqrt{1-\rho(\theta)^2}} - s \| = O(\epsilon + \delta)$, where

$$s \coloneqq \frac{-\rho_\star f(b_\star) + f(b_\star)}{\sqrt{1-\rho_\star^2}} = f(b_\star) \cdot \frac{1-\rho_\star}{\sqrt{1-\rho_\star^2}} = f(b_\star) \cdot \sqrt{\frac{1-\rho_\star}{1+\rho_\star}}$$

and $\rho_\star = \frac{-1+2b_\star - b_\star^2}{1-b_\star^2} = -\frac{1-b_\star}{1+b_\star}$. By plugging in the value of $b_\star$ and $\beta_\star$, we get $s \approx 0.4779$. So we may rewrite

$$\begin{aligned} & W_{i,0} - W_{i,1} \\ &= \sum_{C \in \mathcal{C}_i^+} w(C)\mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_{j_C}^{\perp\{0,i\}} \geq s \pm O(\epsilon + \delta)\} - \sum_{C \in \mathcal{C}_i^-} w(C)\mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_{j_C}^{\perp\{0,i\}} \leq s \pm O(\epsilon + \delta)\} \\ &= \sum_{C \in \mathcal{C}_i} w(C)\mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_{j_C}^{\perp\{0,i\}} \geq s \pm O(\epsilon + \delta)\} - \sum_{C \in \mathcal{C}_i^-} w(C). \end{aligned}$$

In the last step we used the fact that for a continuous random variable X we have $\mathbf{1}\{X \leq s\} = 1 - \mathbf{1}\{X \geq s\}$. In the above calculations, the terms $O(\epsilon + \delta)$ may be different depending on C but we may choose them small enough so that they are all less than $s/2$. Note that $\sum_{C \in \mathcal{C}_i^-} w(C)$ is a constant which does not affect the variance. So, we can apply Lemma 26 and obtain

$$\mathrm{Var}[W_{i,0} - W_{i,1} \mid \mathbf{r} \cdot \mathbf{v}_i^{\perp} = t] = W_i^2 \cdot \Omega(d^{-1}).$$

It follows that

$$\begin{aligned}\mathbb{E}[|W_{i,0} - W_{i,1}| \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp} = t] &\geq \frac{1}{W_i}\cdot\mathbb{E}[(W_{i,0}-W_{i,1})^2 \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}=t]\\ &\geq \frac{1}{W_i}\cdot\mathrm{Var}[W_{i,0}-W_{i,1}\mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}=t]\\ &= W_i\cdot\Omega(d^{-1}).\end{aligned}$$

◀

▶ **Lemma 17.** *For every i and $t \in (f(b_i)-\epsilon, f(b_i)+\epsilon)$ we have $\mathbb{E}\left[W_{i,*} \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}=t\right] = \Theta(\epsilon\cdot W_i)$ where the constant in Θ can be chosen independently from t.*

Proof. Consider a constraint $C \in \mathcal{C}_i$ involving i and some other variable j, represented by the configuration $\theta = (b_i, b_j, b_{ij}, P)$. As computed in the proof of Lemma 16, we have that j is unassigned if and only if

$$\frac{-\rho(\theta)\mathbf{r}\cdot\mathbf{v}_i^{\perp} + f(b_j) - \epsilon}{\sqrt{1-\rho(\theta)^2}} < \mathbf{r}\cdot\mathbf{v}_j^{\perp\{0,i\}} < \frac{-\rho(\theta)\mathbf{r}\cdot\mathbf{v}_i^{\perp} + f(b_j) + \epsilon}{\sqrt{1-\rho(\theta)^2}}.$$

This is an interval of length $\Theta(\epsilon)$. It follows that

$$\mathbb{E}\left[\mathbf{1}\{C\in\mathcal{C}_{i,*}\} \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}=t\right] = \Theta(\epsilon),$$

and therefore

$$\mathbb{E}\left[W_{i,*} \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}=t\right] = \mathbb{E}\left[\sum_{C\in\mathcal{C}} w(C)\mathbf{1}\{C\in\mathcal{C}_{i,*}\} \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}=t\right] = \Theta(\epsilon\cdot W_i).$$

◀

▶ **Lemma 18** (Local gains)**.** *Choose $\epsilon = cd^{-1}\log^{-1}(d)$ for some sufficiently small constant $c>0$. Then, for every variable i, we have $\mathbb{E}[\Delta_i \mid A_*(i)=*] = W_i\cdot\Omega(d^{-1})$.*

Proof. It is sufficient to show that

$$\mathbb{E}[\Delta_i \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}=t] = W_i\cdot\Omega(d^{-1})$$

either for every $t\in(f(b_i)-\epsilon, f(b_i))$ or for every $t\in(f(b_i), f(b_i)+\epsilon)$.

From the definition

$$\begin{aligned}\Delta_i = (W_{i,0}-W_{i,1}-W_{i,*})_+ &\cdot \mathbf{1}\{\mathbf{r}\cdot\mathbf{v}_i^{\perp}\in(f(b_i), f(b_i)+\epsilon)\}\\ + (W_{i,1}-W_{i,0}-W_{i,*})_+ &\cdot \mathbf{1}\{\mathbf{r}\cdot\mathbf{v}_i^{\perp}\in(f(b_i)-\epsilon, f(b_i))\}\end{aligned}$$

we have

$$\begin{aligned}&\mathbb{E}[\Delta_i \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}\in(f(b_i), f(b_i)+\epsilon)]\\ &= \mathbb{E}\left[(W_{i,0}-W_{i,1}-W_{i,*})_+ \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}\in(f(b_i), f(b_i)+\epsilon)\right]\\ &\geq \mathbb{E}\left[(W_{i,0}-W_{i,1})_+ - W_{i,*} \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}\in(f(b_i), f(b_i)+\epsilon)\right]\\ &= \mathbb{E}\left[\frac{|W_{i,0}-W_{i,1}| + (W_{i,0}-W_{i,1})}{2} - W_{i,*} \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}\in(f(b_i), f(b_i)+\epsilon)\right],\end{aligned}$$

and similarly

$$\begin{aligned}&\mathbb{E}[\Delta_i \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}\in(f(b_i)-\epsilon, f(b_i))]\\ &\geq \mathbb{E}\left[\frac{|W_{i,1}-W_{i,0}| + (W_{i,1}-W_{i,0})}{2} - W_{i,*} \mid \mathbf{r}\cdot\mathbf{v}_i^{\perp}\in(f(b_i)-\epsilon, f(b_i))\right].\end{aligned}$$

Assume that $\mathbb{E}[W_{i,0} - W_{i,1} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i), f(b_i) + \epsilon)] \geq -W_i/(d\sqrt{\log d})$. Then by Lemma 16 and Lemma 17 we have

$$\begin{aligned}
&\mathbb{E}\left[\frac{|W_{i,0} - W_{i,1}| + (W_{i,0} - W_{i,1})}{2} - W_{i,*} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i), f(b_i) + \epsilon)\right] \\
&\geq W_i \cdot \left(\Omega(d^{-1}) - d^{-1}\log^{-1/2}(d) - O(d^{-1}\log^{-1}(d))\right) \\
&= W_i \cdot \Omega(d^{-1}).
\end{aligned}$$

Otherwise, we have $\mathbb{E}[W_{i,0} - W_{i,1} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i), f(b_i) + \epsilon)] < -W_i/(d\sqrt{\log d})$, or equivalently $\mathbb{E}[W_{i,1} - W_{i,0} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i), f(b_i) + \epsilon)] > W_i/(d\sqrt{\log d})$. Observe that by Equation (4.1) the function $\mathbb{E}[W_{i,1} - W_{i,0} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp = t]$ is $O(W_i)$-Lipschitz in t, so we have

$$\mathbb{E}[W_{i,1} - W_{i,0} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i) - \epsilon, f(b_i))] > W_i/(d\sqrt{\log d}) - \epsilon \cdot O(W_i) = W_i \cdot \Omega(d^{-1}\log^{-1/2}(d)).$$

It follows that

$$\begin{aligned}
&\mathbb{E}\left[\frac{|W_{i,1} - W_{i,0}| + (W_{i,1} - W_{i,0})}{2} - W_{i,*} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i) - \epsilon, f(b_i))\right] \\
&\geq W_i \cdot \left(\Omega(d^{-1}) + \Omega(d^{-1}\log^{-1/2}(d)) - O(d^{-1}\log^{-1}(d))\right) \\
&= W_i \cdot \Omega(d^{-1}).
\end{aligned}$$

Thus, we conclude that $\mathbb{E}[\Delta_i \mid A_*(i) = *] = W_i \cdot \Omega(d^{-1})$. ◀

We are now ready to prove Theorem 14.

Proof of Theorem 14. By Lemma 10, we have

$$\begin{aligned}
\mathbb{E}[\mathrm{Val}(I, A)] &\geq \mathsf{Soundness}_f(\mathcal{D}) + \sum_{i \in V} \mathbb{E}[\Delta_i] \\
&\geq \mathsf{Soundness}_f(\mathcal{D}) + \sum_{i \in V} \mathbb{E}[\Delta_i \mid A_*(i) = *] \cdot \Pr[A_*(i) = *] \\
&\geq \mathsf{Soundness}_f(\mathcal{D}) + \sum_{i \in V} W_i \cdot \Omega(d^{-1}) \cdot \Omega(\epsilon) \\
&\geq \mathsf{Soundness}_f(\mathcal{D}) + \Omega(d^{-2}\log^{-1} d).
\end{aligned}$$

Here we used the assumption that $\sum_{i \in V} W_i = 2$ (since total weight is 1 and each constraint is counted twice) and Lemma 18. ◀

5 Analysis for General Boolean MAX 2-CSPs

In this section we analyze general Boolean MAX 2-CSPs. To this end we need the distribution of configurations to be *reasonable* as introduced below.

▶ **Definition 19.** *We say that two predicates $P_1, P_2 : \{-1,1\}^2 \to \{0,1\}$ are* aligned *if there exist $\sigma_1, \sigma_2 \in \{-1,1\}$ and $s \in \{0,1\}$ such that*

$$P_1(-1, \sigma_1) = P_2(-\sigma_2, \sigma_1\sigma_2) = s \quad \text{and} \quad P_1(1, \sigma_1) = P_2(\sigma_2, \sigma_1\sigma_2) = 1 - s.$$

We say that they are anti-aligned *if there exist $\sigma_1, \sigma_2 \in \{-1,1\}$ and $s \in \{0,1\}$ such that*

$$P_1(-1, \sigma_1) = P_2(\sigma_2, \sigma_1\sigma_2) = s \quad \text{and} \quad P_1(1, \sigma_1) = P_2(-\sigma_2, \sigma_1\sigma_2) = 1 - s.$$

Informally, if we look at two constraints with one variable in common, then they are aligned if there is a way to fix the other variables to induce the same unary preference on the shared variable, and they are anti-aligned if they induce opposite unary preferences.

▶ **Definition 20.** *We say that a distribution of configurations $\mathcal{D}$ representing some SDP solution is* reasonable *if the following conditions hold:*

- *For every $\theta = (b_i, b_j, b_{ij}, P) \in \mathrm{supp}(\mathcal{D})$, we have $|b_i|, |b_j| \le 0.4$ and $|b_{ij}| \in [0.5, 0.9]$.*
- *If $\theta_1 = (b_i, b_j, b_{ij}, P_1), \theta_2 = (b_i, b_k, b_{ik}, P_2) \in \mathrm{supp}(\mathcal{D})$ represent some constraints (P_1, i, j) and (P_2, i, k), then $\rho(\theta_1)\rho(\theta_2) \ge 0$ if P_1 and P_2 are aligned and $\rho(\theta_1)\rho(\theta_2) \le 0$ if P_1 and P_2 are anti-aligned.*

▶ Remark 21. The conditions for reasonableness may seem somewhat arbitrary. Indeed, the constants are chosen only for convenience: with these values we are able to prove bounds on the covariances as in Lemma 22. That said, it may be verified that the hardest (optimal under UGC) distributions for MAX CUT and MAX 2-SAT are both reasonable. Furthermore, the known (numerically) hardest distributions (though not known to be optimal) for MAX 2-AND and MAX DI-CUT due to [7] are also reasonable.

The main benefit of reasonableness is the following bound on covariances in a local neighborhood. The proof of this lemma is deferred to the end of this section.

▶ **Lemma 22.** *Let $\mathcal{D}$ be a reasonable distribution of configurations representing some SDP solution $\{\mathbf{v}_i \mid i \in \{0\} \cup [n]\}$ which satisfies all triangle inequalities. Let $\theta_1 = (b_i, b_j, b_{ij}, P_1), \theta_2 = (b_i, b_k, b_{ik}, P_2) \in \mathrm{supp}(\mathcal{D})$ be two configurations representing some constraints (P_1, i, j) and (P_2, i, k). If P_1 and P_2 are aligned, then $\mathbf{v}_j^{\perp\{0,i\}} \cdot \mathbf{v}_k^{\perp\{0,i\}} \ge -5/6$. If P_1 and P_2 are anti-aligned, then $\mathbf{v}_j^{\perp\{0,i\}} \cdot \mathbf{v}_k^{\perp\{0,i\}} \le 5/6$.*

Under the reasonableness condition, we prove the following theorem.

▶ **Theorem 23.** *Let $\mathcal{P}$ be a set of Boolean predicates of arity 2 such that every $P \in \mathcal{P}$ depends on both coordinates. Let $d \in \mathbb{N}$, and $B > 0$. Let I be an instance of* MAX CSP($\mathcal{P}$) *such that every variable appears in at most d constraints. Let $\mathcal{D}$ be a reasonable distribution of configurations corresponding to an SDP solution satisfying all triangle inequalities. Then, for some $\epsilon = \Theta(d^{-2}\log^{-2}(d))$ and any $f : [-1, 1] \to [-B, B]$, Algorithm 1 returns an assignment to I that satisfies at least $\mathsf{Soundness}_f(\mathcal{D}) + \Omega(d^{-4}\log^{-7/2} d)$ fraction of constraints, where the constant inside the Ω notation depends only on $\mathcal{P}$ and B.*

We will implicitly assume the assumptions in Theorem 23 in our analysis for the remainder of the section. We use A_* to denote the (partial) assignment A before the execution of the for-loop at Line 8. The overall idea is similar to Theorem 14: we show that $W_{i,0} - W_{i,1}$ has a large variance which overwhelms the expected value of $W_{i,*}$, and therefore at least one of the two terms in Δ_i will have a nontrivial contribution. The variance lower bound for $W_{i,0} - W_{i,1}$ is shown in the following lemma.

▶ **Lemma 24.** *There exist $c, \epsilon_0 > 0$ (depending only on the configurations and the threshold function f) such that for any $\epsilon < \epsilon_0$, $i \in V$ and any $t \in (f(b_i) - \epsilon, f(b_i) + \epsilon)$, we have*

$$\mathbb{E}[|W_{i,0} - W_{i,1}| \mid \mathbf{r} \cdot \mathbf{v}_i^{\perp} = t] \ge \frac{c \cdot W_i}{d^2 \log^{3/2} d}.$$

Proof. We have

$$W_{i,0} - W_{i,1} = \sum_{C \in \mathcal{C}_i} w(C) \left(\mathbf{1}\{C \in \mathcal{C}_{i,0}\} - \mathbf{1}\{C \in \mathcal{C}_{i,1}\}\right).$$

Let $C \in \mathcal{C}_i$ be a constraint on i and some other variable j, represented by the configuration $\theta = (b_i, b_j, b_{ij}, P)$. Observe that since by assumption P depends on both variables, the indicator functions $\mathbf{1}\{C \in \mathcal{C}_{i,0}\}$ and $\mathbf{1}\{C \in \mathcal{C}_{i,1}\}$ are two distinct functions among the three functions $\mathbf{1}\{A_*(j) = \texttt{true}\}$, $\mathbf{1}\{A_*(j) = \texttt{false}\}$, and 0 (as a constant function). Conditioned on $\mathbf{r} \cdot \mathbf{v}_i^{\perp} = t$, we may write the first two indicators as $\mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}} \geq s_j\}$, $\mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}} \leq s'_j\}$, where s_j, s'_j are some thresholds whose absolute values are bounded due to our reasonableness assumption (the expressions for these thresholds are the same as those in (4.1)). By using the fact that $\mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}} \leq s'_j\} = 1 - \mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}} \geq s'_j\}$, we may therefore rewrite (5) as

$$W_{i,0} - W_{i,1} = \sum_{j \in J_1} w_j \mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}} \geq s_j\} - \sum_{j \in J_2} w_j \mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}} \geq s'_j\} + c$$

where J_1, J_2 are some (multi-)sets of variables and c is some constant. We also have $\sum_{j \in J_1} w_j + \sum_{j \in J_2} w_j \in [W_i, 2W_i]$ since each constraint contributes either 1 or 2 nonzero indicator functions.

Now we aim to bound the inner products between the $\mathbf{v}_j^{\perp\{0,i\}}$ vectors, which are equivalently the covariances between the standard Gaussian variables $\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}}$. Take another constraint C' involving i and some third variable k, represented by $\theta' = (b_i, b_k, b_{ik}, P')$. Then P and P' are aligned if there exist $s \in \{0,1\}$ and $\mathtt{s} \in \{\texttt{true}, \texttt{false}\}$ such that

$$\mathbf{1}\{C \in \mathcal{C}_{i,s}\} = \mathbf{1}\{A_*(j) = \mathtt{s}\} \quad \text{and} \quad \mathbf{1}\{C' \in \mathcal{C}_{i,s}\} = \mathbf{1}\{A_*(k) = \mathtt{s}\},$$

or

$$\mathbf{1}\{C \in \mathcal{C}_{i,s}\} = \mathbf{1}\{A_*(j) = \mathtt{s}\} \quad \text{and} \quad \mathbf{1}\{C' \in \mathcal{C}_{i,1-s}\} = \mathbf{1}\{A_*(k) = \texttt{not } \mathtt{s}\},$$

where we used $\texttt{not } \mathtt{s}$ to mean the truth value other than $\mathtt{s}$. These are exactly the cases where we get both $j, k \in J_1$ or $j, k \in J_2$. By Lemma 22, we have $\mathbf{v}_j^{\perp\{0,i\}} \cdot \mathbf{v}_k^{\perp\{0,i\}} \geq -5/6$ if P and P' are aligned. On the other hand, P and P' are anti-aligned if there exist $s \in \{0,1\}$ and $\mathtt{s} \in \{\texttt{true}, \texttt{false}\}$ such that

$$\mathbf{1}\{C \in \mathcal{C}_{i,s}\} = \mathbf{1}\{A_*(j) = \mathtt{s}\} \quad \text{and} \quad \mathbf{1}\{C' \in \mathcal{C}_{i,s}\} = \mathbf{1}\{A_*(k) = \texttt{not } \mathtt{s}\},$$

or

$$\mathbf{1}\{C \in \mathcal{C}_{i,s}\} = \mathbf{1}\{A_*(j) = \mathtt{s}\} \quad \text{and} \quad \mathbf{1}\{C' \in \mathcal{C}_{i,1-s}\} = \mathbf{1}\{A_*(k) = \mathtt{s}\}.$$

These are exactly the cases where one of $\{j, k\}$ is in J_1 and the other is in J_2. Again by Lemma 22, we have $\mathbf{v}_j^{\perp\{0,i\}} \cdot \left(-\mathbf{v}_k^{\perp\{0,i\}}\right) \geq -5/6$ if P and P' are anti-aligned.

Hence, by rewriting

$$\begin{aligned} W_{i,0} - W_{i,1} &= \sum_{j \in J_1} w_j \mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}} \geq s_j\} - \sum_{j \in J_2} w_j \mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}} \geq s'_j\} + c \\ &= \sum_{j \in J_1} w_j \mathbf{1}\{\mathbf{r} \cdot \mathbf{v}_j^{\perp\{0,i\}} \geq s_j\} + \sum_{j \in J_2} w_j \mathbf{1}\{\mathbf{r} \cdot (-\mathbf{v}_j^{\perp\{0,i\}}) \geq -s'_j\} - \sum_{j \in J_2} w_j + c, \end{aligned}$$

we obtain an expression which is the weighted sum of indicators of the form $\mathbf{1}\{g_j \geq s''_j\}$ (plus some constant) where g_j's are correlated standard Gaussians and the covariances between them are all at least $-5/6$. We can therefore apply Lemma 28, which gives us

$$\mathrm{Var}[W_{i,0} - W_{i,1} \mid \mathbf{r} \cdot \mathbf{v}_i^{\perp} = t] \geq \Omega\left(\frac{W_i^2}{d^2 \log^{3/2} d}\right).$$

Since $|W_{i,0} - W_{i,1}| \le W_i$, we have

$$\mathbb{E}[|W_{i,0} - W_{i,1}| \mid \mathbf{r} \cdot \mathbf{v}_i^\perp = t] \ge \frac{1}{W_i}\,\mathbb{E}[(W_{i,0} - W_{i,1})^2 \mid \mathbf{r} \cdot \mathbf{v}_i^\perp = t] \ge \Omega\left(\frac{W_i}{d^2 \log^{3/2} d}\right). \quad \blacktriangleleft$$

▶ **Lemma 25** (Local gains). *Choose $\epsilon = cd^{-2}\log^{-2}(d)$ for some sufficiently small constant $c > 0$. Then, for every variable i, we have $\mathbb{E}[\Delta_i \mid A_*(i) = *] = W_i \cdot \Omega(d^{-2}\log^{-3/2}(d))$.*

Proof. As in the proof of Lemma 18, it is sufficient to show that

$$\mathbb{E}[\Delta_i \mid \mathbf{r} \cdot \mathbf{v}_i^\perp = t] = W_i \cdot \Omega(d^{-2}\log^{-3/2}(d))$$

either for every $t \in (f(b_i) - \epsilon, f(b_i))$ or for every $t \in (f(b_i), f(b_i) + \epsilon)$.

As in Lemma 17, we have $\mathbb{E}\left[W_{i,*} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp = t\right] = \Theta(\epsilon \cdot W_i)$. Assume that $\mathbb{E}[W_{i,0} - W_{i,1} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i), f(b_i) + \epsilon)] \ge -W_i \cdot d^{-2} \cdot \log^{-2}(d)$. Then by Lemma 24 we have

$$\begin{aligned}
&\mathbb{E}[\Delta_i \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i), f(b_i) + \epsilon)] \\
&\ge \mathbb{E}\left[\frac{|W_{i,0} - W_{i,1}| + (W_{i,0} - W_{i,1})}{2} - W_{i,*} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i), f(b_i) + \epsilon)\right] \\
&\ge W_i \cdot \Big(\Omega(d^{-2}\log^{-3/2}(d)) - d^{-2}\log^{-2}(d) - O(d^{-2}\log^{-2}(d))\Big) \\
&= W_i \cdot \Omega(d^{-2}\log^{-3/2}(d)).
\end{aligned}$$

Otherwise, we have $\mathbb{E}[W_{i,0} - W_{i,1} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i), f(b_i) + \epsilon)] < -W_i \cdot d^{-2} \cdot \log^{-2}(d)$, or equivalently $\mathbb{E}[W_{i,1} - W_{i,0} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i), f(b_i) + \epsilon)] > W_i \cdot d^{-2} \cdot \log^{-2}(d)$. Since the function $\mathbb{E}[W_{i,1} - W_{i,0} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp = t]$ is $O(W_i)$-Lipschitz in t, we have

$$\mathbb{E}[W_{i,1} - W_{i,0} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i) - \epsilon, f(b_i))] > W_i \cdot d^{-2}\log^{-2}(d) - \epsilon \cdot O(W_i) \ge -W_i \cdot O(d^{-2}\log^{-2}(d)).$$

It follows that

$$\begin{aligned}
&\mathbb{E}[\Delta_i \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i) - \epsilon, f(b_i))] \\
&\ge \mathbb{E}\left[\frac{|W_{i,1} - W_{i,0}| + (W_{i,1} - W_{i,0})}{2} - W_{i,*} \mid \mathbf{r} \cdot \mathbf{v}_i^\perp \in (f(b_i) - \epsilon, f(b_i))\right] \\
&\ge W_i \cdot \Big(\Omega(d^{-2}\log^{-3/2}(d)) - O(d^{-2}\log^{-2}(d)) - O(d^{-2}\log^{-2}(d))\Big) \\
&= W_i \cdot \Omega(d^{-2}\log^{-3/2}(d)).
\end{aligned}$$

Thus, we conclude $\mathbb{E}[\Delta_i \mid A_*(i) = *] = W_i \cdot \Omega(d^{-2}\log^{-3/2}(d))$. $\blacktriangleleft$

Proof of Theorem 23. Follows similarly as in the proof of Theorem 14. By Lemma 10, we have

$$\begin{aligned}
\mathbb{E}[\mathrm{Val}(I, A)] &\ge \mathsf{Soundness}_f(\mathcal{D}) + \sum_{i \in V} \mathbb{E}[\Delta_i] \\
&\ge \mathsf{Soundness}_f(\mathcal{D}) + \sum_{i \in V} \mathbb{E}[\Delta_i \mid A_*(i) = *] \cdot \Pr[A_*(i) = *] \\
&\ge \mathsf{Soundness}_f(\mathcal{D}) + \sum_{i \in V} W_i \cdot \Omega(d^{-2}\log^{-3/2}(d)) \cdot \Omega(\epsilon) \\
&\ge \mathsf{Soundness}_f(\mathcal{D}) + \Omega(d^{-4}\log^{-7/2} d).
\end{aligned}$$

$\blacktriangleleft$

5.1 Proof of Lemma 22

We only prove the case where P_1 and P_2 are aligned since the anti-aligned case can be reduced to the aligned case by negating the variable k as well as $\mathbf{v}_k$. Assume that P_1 and P_2 are aligned. Then, since $\mathcal{D}$ is reasonable, we have $\rho(\theta_1)\rho(\theta_2) \geq 0$, $|b_i|, |b_j|, |b_k| \leq 0.4$ and $|b_{ij}|, |b_{ik}| \geq 0.5$. This implies that $|b_{ij}| \geq |b_i||b_j|$, and therefore b_{ij} and $\rho(\theta_1) = \frac{b_{ij}-b_ib_j}{\sqrt{(1-b_i^2)(1-b_j^2)}}$ have the same sign. We may then by possibly negating $\mathbf{v}_i$ assume that $b_{ij}, b_{ik}, \rho(\theta_1), \rho(\theta_2) \geq 0$.

For $\ell \in \{j,k\}$, let $\mathbf{u}_\ell^{\perp\{0,i\}}$ be the component of $\mathbf{v}_\ell$ parallel to $\mathbf{v}_\ell^{\perp\{0,i\}}$, so we have

$$\mathbf{u}_\ell^{\perp\{0,i\}} = \sqrt{1-b_\ell^2}\mathbf{v}_\ell^\perp - \frac{b_{i\ell}-b_ib_\ell}{\sqrt{1-b_i^2}}\mathbf{v}_i^\perp.$$

It follows that

$$\|\mathbf{u}_\ell^{\perp\{0,i\}}\|^2 = 1 - b_\ell^2 - \frac{(b_{i\ell}-b_ib_\ell)^2}{1-b_i^2}$$

for $\ell \in \{j,k\}$, and

$$\mathbf{u}_j^{\perp\{0,i\}} \cdot \mathbf{u}_k^{\perp\{0,i\}} = (b_{jk}-b_jb_k) - \frac{(b_{ij}-b_ib_j)(b_{ik}-b_ib_k)}{1-b_i^2}.$$

For $\sigma \in \{-1,1\}$ and $\ell \in \{j,k\}$, let us define

$$p_\sigma = \frac{1+\sigma b_i}{2}, \quad \text{and} \quad q_{\sigma,\ell} = \frac{1+\sigma b_i - \sigma b_\ell - b_{i\ell}}{2(1+\sigma b_i)}.$$ [4]

By the triangle inequalities, we have $1 - b_i + b_\ell - b_{i\ell} \geq 0$, so $b_\ell \geq -1 + b_i + b_{i\ell} \geq -0.5 + b_i$. Since $|b_\ell| \leq 0.4$, we have $b_\ell \geq \max(-0.4, -0.5+b_i)$. So we have

$$q_{+1,\ell} = \frac{1+b_i-b_\ell-b_{i\ell}}{2(1+b_i)} \leq \frac{1+b_i-\max(-0.4,-0.5+b_i)-0.5}{2(1+b_i)}.$$

When $b_i \geq 0.1$, the right hand side above is $\frac{1}{2(1+b_i)} \leq \frac{5}{11}$. When $b_i \leq 0.1$, the right hand side above is $\frac{0.9+b_i}{2(1+b_i)} = \frac{1}{2} - \frac{0.1}{2(1+b_i)} \leq \frac{5}{11}$. So in either case we obtain $q_{+1,\ell} \leq \frac{5}{11}$. Similarly, for $q_{-1,\ell}$ we use the triangle inequality $1+b_i-b_\ell-b_{i\ell} \geq 0$, from which we deduce $b_\ell \leq \min(0.4, 0.5+b_i)$, and therefore

$$q_{-1,\ell} = \frac{1-b_i+b_\ell-b_{i\ell}}{2(1-b_i)} \leq \frac{1-b_i+\min(0.4,0.5+b_i)-0.5}{2(1-b_i)}.$$

When $b_i \leq -0.1$, the right hand side above is $\frac{1}{2(1-b_i)} \leq \frac{5}{11}$. When $b_i \geq -0.1$, the right hand side above is $\frac{0.9-b_i}{2(1-b_i)} = \frac{1}{2} - \frac{0.1}{2(1-b_i)} \leq \frac{5}{11}$. So in either case we obtain $q_{-1,\ell} \leq \frac{5}{11}$.

Now let's relate the quantities $p_\sigma, q_{\sigma,\ell}$ to the SDP vectors. We have

$$\begin{aligned}
\sum_{\sigma\in\{-1,1\}} p_\sigma q_{\sigma,\ell}(1-q_{\sigma,\ell}) &= \sum_{\sigma\in\{-1,1\}} \frac{1+\sigma b_i}{2} \cdot \frac{(1+\sigma b_i)^2 - (\sigma b_\ell + b_{i\ell})^2}{4(1+\sigma b_i)^2} \\
&= \frac{1}{4}\left(1 - \frac{(b_\ell+b_{i\ell})^2}{2(1+b_i)} - \frac{(b_\ell-b_{i\ell})^2}{2(1-b_i)}\right) \\
&= \frac{1}{4}\|\mathbf{u}_\ell^{\perp\{0,i\}}\|^2,
\end{aligned}$$

as well as

$$\begin{aligned}
&\sum_{\sigma\in\{-1,1\}} p_\sigma q_{\sigma,j} q_{\sigma,k} \\
&= \sum_{\sigma\in\{-1,1\}} \frac{(1+\sigma b_i - \sigma b_j - b_{ij})(1+\sigma b_i - \sigma b_k - b_{ik})}{8(1+\sigma b_i)} \\
&= \sum_{\sigma\in\{-1,1\}} \frac{1}{4}\left(\frac{1+\sigma b_i - (\sigma b_j + b_{ij} + \sigma b_k + b_{ik})}{2} + \frac{(b_j+\sigma b_{ij})(b_k+\sigma b_{ik})}{2(1+\sigma b_i)}\right) \\
&= \frac{1}{4}\left(1 - b_{ij} - b_{ik} + \frac{(b_j+b_{ij})(b_k+b_{ik})}{2(1+b_i)} + \frac{(b_j-b_{ij})(b_k-b_{ik})}{2(1-b_i)}\right) \\
&= \frac{1}{4}\left(1 - b_{ij} - b_{ik} + b_j b_k + \frac{(b_{ij}-b_i b_j)(b_{ik}-b_i b_k)}{1-b_i^2}\right) \\
&= \frac{1}{4}\left(1 - b_{ij} - b_{ik} + b_{jk} - \mathbf{u}_j^{\perp\{0,i\}} \cdot \mathbf{u}_k^{\perp\{0,i\}}\right) \\
&\geq -\frac{1}{4}\mathbf{u}_j^{\perp\{0,i\}} \cdot \mathbf{u}_k^{\perp\{0,i\}}.
\end{aligned}$$

Here the last inequality is due to the triangle inequalities.

Observe that since $q_{\sigma,\ell} \leq \alpha$ where $\alpha = \frac{5}{11}$ for every $\sigma \in \{-1,1\}, \ell \in \{j,k\}$, we have

$$q_{\sigma,j} q_{\sigma,k} \leq \frac{\alpha}{1-\alpha}\sqrt{q_{\sigma,j}(1-q_{\sigma,j}) q_{\sigma,k}(1-q_{\sigma,k})}$$

and therefore

$$\begin{aligned}
&\sum_{\sigma\in\{-1,1\}} p_\sigma q_{\sigma,j} q_{\sigma,k} \\
&\leq \frac{\alpha}{1-\alpha} \cdot \sum_{\sigma\in\{-1,1\}} p_\sigma \sqrt{q_{\sigma,j}(1-q_{\sigma,j}) q_{\sigma,k}(1-q_{\sigma,k})} \\
&\leq \frac{\alpha}{1-\alpha} \cdot \sqrt{\left(\sum_{\sigma\in\{-1,1\}} p_\sigma q_{\sigma,j}(1-q_{\sigma,j})\right)} \sqrt{\left(\sum_{\sigma\in\{-1,1\}} p_\sigma q_{\sigma,k}(1-q_{\sigma,k})\right)},
\end{aligned}$$

where the last inequality is due to Cauchy-Schwarz. Thus, we have

$$\begin{aligned}
&\mathbf{u}_j^{\perp\{0,i\}} \cdot \mathbf{u}_k^{\perp\{0,i\}} \\
&\geq -4 \sum_{\sigma\in\{-1,1\}} p_\sigma q_{\sigma,j} q_{\sigma,k} \\
&\geq -4 \cdot \frac{\alpha}{1-\alpha} \cdot \sqrt{\left(\sum_{\sigma\in\{-1,1\}} p_\sigma q_{\sigma,j}(1-q_{\sigma,j})\right)} \sqrt{\left(\sum_{\sigma\in\{-1,1\}} p_\sigma q_{\sigma,k}(1-q_{\sigma,k})\right)} \\
&= -\frac{\alpha}{1-\alpha} \cdot \|\mathbf{u}_j^{\perp\{0,i\}}\| \|\mathbf{u}_k^{\perp\{0,i\}}\|
\end{aligned}$$

By shifting the factors, we obtain

$$\mathbf{v}_j^{\perp\{0,i\}} \cdot \mathbf{v}_k^{\perp\{0,i\}} = \frac{\mathbf{u}_j^{\perp\{0,i\}} \cdot \mathbf{u}_k^{\perp\{0,i\}}}{\|\mathbf{u}_j^{\perp\{0,i\}}\| \|\mathbf{u}_k^{\perp\{0,i\}}\|} \geq -\frac{\alpha}{1-\alpha} = -\frac{5}{6}.$$

This completes the proof.

References

1 Per Austrin. Balanced MAX 2-SAT Might Not Be the Hardest. In *Symposium on Theory of Computing (STOC)*, pages 189–197, 2007. doi:10.1145/1250790.1250818.

2 Per Austrin. Towards sharp inapproximability for any 2-csp. *SIAM Journal on Computing*, 39(6):2430–2463, 2010. doi:10.1137/070711670.

3 Per Austrin, Siavosh Benabbas, and Konstantinos Georgiou. Better balance by being biased: A 0.8776-approximation for max bisection. *ACM Transactions on Algorithms (TALG)*, 13(1):1–27, 2016. doi:10.1145/2907052.

4 Per Austrin, Subhash Khot, and Muli Safra. Inapproximability of vertex cover and independent set in bounded degree graphs. In *2009 24th Annual IEEE Conference on Computational Complexity*, pages 74–80. IEEE, 2009. doi:10.1109/CCC.2009.38.

5 Boaz Barak, Ankur Moitra, Ryan O'Donnell, Prasad Raghavendra, Oded Regev, David Steurer, Luca Trevisan, Aravindan Vijayaraghavan, David Witmer, and John Wright. Beating the Random Assignment on Constraint Satisfaction Problems of Bounded Degree. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pages 110–123, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.110.

6 Piotr Berman and Marek Karpinski. On some tighter inapproximability results. In *Automata, Languages and Programming: 26th International Colloquium, ICALP'99 Prague, Czech Republic, July 11–15, 1999 Proceedings*, pages 200–209. Springer, 2002.

7 Joshua Brakensiek, Neng Huang, Aaron Potechin, and Uri Zwick. Separating MAX 2-AND, MAX DI-CUT and MAX CUT. In *Symposium on Foundations of Computer Science (FOCS)*, pages 234–252, 2023. doi:10.1109/FOCS57990.2023.00023.

8 Joshua Brakensiek, Neng Huang, and Uri Zwick. Tight Approximability of MAX 2-SAT and Relatives, Under UGC. In *Symposium on Discrete Algorithms (SODA)*, pages 1328–1344, 2024. doi:10.1137/1.9781611977912.53.

9 Uriel Feige, Marek Karpinski, and Michael Langberg. Improved approximation of max-cut on graphs of bounded degree. *Journal of Algorithms*, 43(2):201–219, 2002. doi:10.1016/S0196-6774(02)00005-6.

10 Uriel Feige and Gideon Schechtman. On the optimality of the random hyperplane rounding technique for max cut. *Random Structures & Algorithms*, 20(3):403–440, 2002. doi:10.1002/RSA.10036.

11 Mikael Florén. Approximation of max-cut on graphs of bounded degree, 2016.

12 Michel X. Goemans and David P. Williamson. .879-approximation algorithms for MAX CUT and MAX 2sat. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 422–431. ACM, 1994. doi:10.1145/195058.195216.

13 Eran Halperin, Dror Livnat, and Uri Zwick. Max cut in cubic graphs. *Journal of Algorithms*, 53(2):169–185, 2004. doi:10.1016/J.JALGOR.2004.06.001.

14 Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001. doi:10.1145/502090.502098.

15 Jun-Ting Hsieh and Pravesh K. Kothari. Approximating Max-Cut on Bounded Degree Graphs: Tighter Analysis of the FKL Algorithm. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 77:1–77:7, 2023. doi:10.4230/LIPIcs.ICALP.2023.77.

16 Subhash Khot. On the Power of Unique 2-Prover 1-Round Games. In *Symposium on Theory of Computing (STOC)*, pages 767–775, 2002. doi:10.1145/509907.510017.

17 Subhash Khot. Guest column: inapproximability results via long code based pcps. *ACM SIGACT News*, 36(2):25–42, 2005. doi:10.1145/1067309.1067318.

18 Subhash Khot. On the unique games conjecture (invited survey). In *2010 IEEE 25th annual conference on computational complexity*, pages 99–121. IEEE Computer Society, 2010. doi:10.1109/CCC.2010.19.

19 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? *SIAM Journal on Computing (SICOMP)*, 37(1):319–357, 2007. doi:10.1137/S0097539705447372.

20 Michael Lewin, Dror Livnat, and Uri Zwick. Improved Rounding Techniques for the MAX 2-SAT and MAX DI-CUT Problems. In *International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 67–82, 2002. doi:10.1007/3-540-47867-1_6.

21 Konstantin Makarychev and Yury Makarychev. Approximation algorithms for csps. In *Dagstuhl Follow-Ups*, volume 7. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/DFU.VOL7.15301.11.

22 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

23 Prasad Raghavendra. Optimal Algorithms and Inapproximability Results for Every CSP? In *Symposium on Theory of Computing (STOC)*, pages 245–254, 2008. doi:10.1145/1374376.1374414.

A Variance of Sums of Gaussian Threshold Indicator Functions

In this appendix, we study the variance of functions of the form

$$\sum_{i=1}^{d} w_i \mathbf{1}\{g_i \geq s_i\},$$

where $(g_1, \ldots, g_d)$ is a centered Gaussian vector with unit variances, $s_1, \ldots, s_d \in \mathbb{R}$ are some thresholds for the Gaussians, and $w_1, \ldots, w_d \geq 0$ are some nonzero weights. We state a few different versions with different assumptions on the covariance structure or on the thresholds. They will be used to show that there is sufficient variance in a local neighborhood of a bounded-degree MAX 2-CSP instance for an algorithm to exploit, generalizing Lemma 7 from [15] which served a similar purpose for MAX CUT. Due to the page limit, we defer the proofs in this section to the full version of the paper.

In the first version, there is no assumption on the covariance structure but the thresholds are assumed to be close to some nonzero real number.

▶ **Lemma 26.** *Let $(g_1, \ldots, g_d)$ be a centered Gaussian vector with unit variances. Let $s \in \mathbb{R}$ be nonzero, $\delta = |s|/2$, $s_1, \ldots, s_d \in (s - \delta, s + \delta)$, and $w_1, \ldots, w_d \in \mathbb{R}^{\geq 0}$, then we have*

$$\operatorname{Var}\left[\sum_{i=1}^{d} w_i \cdot \mathbf{1}\{g_i \geq s_i\}\right] = \left(\sum_{i=1}^{d} w_i\right)^2 \cdot \Omega(d^{-1}),$$

where the constant inside Ω depends only on s.

In the second version we assume that the covariances are somewhat bounded away from 1 in absolute value, and that the thresholds are also absolutely bounded.

▶ **Lemma 27.** *Fix some constants $B > 0$. Let $(g_1, \ldots, g_d)$ be a centered Gaussian vector with unit variances and pairwise covariances in $\left[-1 + \frac{1}{d^2 \log^2 d}, 1 - \frac{1}{d^2 \log^2 d}\right]$. Let $s_1, \ldots, s_d \in [-B, B]$ and $w_1, \ldots, w_d \in \mathbb{R}^{\geq 0}$, then we have*

$$\operatorname{Var}\left[\sum_{i=1}^{d} w_i \cdot \mathbf{1}\{g_i \geq s_i\}\right] = \left(\sum_{i=1}^{d} w_i\right)^2 \cdot \Omega(d^{-2} \log^{-3/2} d),$$

where the constant inside Ω depends only on B.

In the last version, we assume that the covariances are bounded away from -1 by a constant, and again the thresholds are bounded in absolute value.

▶ **Lemma 28.** *Fix some constants $\eta, B > 0$. Let $(g_1, \ldots, g_d)$ be a centered Gaussian vector with unit variances and pairwise covariances at least $-1 + \eta$. Let $s_1, \ldots, s_d \in [-B, B]$ and $w_1, \ldots, w_d \in \mathbb{R}^{\geq 0}$, then we have*

$$\operatorname{Var}\left[\sum_{i=1}^{d} w_i \cdot \mathbf{1}\{g_i \geq s_i\}\right] = \left(\sum_{i=1}^{d} w_i\right)^2 \cdot \Omega(d^{-2} \log^{-3/2} d),$$

where the constant inside Ω depends only on η and B.

Incremental Consistent k-Center Clustering

Mara Grilnberger
Department of Computer Science, University of Salzburg, Austria

Antonis Skarlatos
Department of Computer Science, University of Warwick, Coventry, UK

Abstract

In the incremental consistent k-center clustering problem, we are given a sequence of adversarial point insertions and aim to maintain a k-center solution with small approximation ratio and small recourse. In this work, we explore the following question: what is the best approximation ratio of a polynomial-time algorithm with a worst-case recourse of 1? Our result improves upon the 6-approximation algorithm of Forster and Skarlatos [SODA '25], which itself improved over the 8-approximation algorithm of Charikar, Chekuri, Feder, and Motwani [STOC '97]. Moreover, we show that any incremental k-center algorithm that achieves an approximation ratio strictly less than $\sqrt{2}$ requires a worst-case recourse of k.

2012 ACM Subject Classification Theory of computation → Facility location and clustering

Keywords and phrases Consistent Clustering, k-Center, Dynamic Algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.29

Category APPROX

Funding *Mara Grilnberger*: This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 947702). This publication has been supported by the EXDIGIT (Excellence in Digital Sciences and Interdisciplinary Technologies) project, funded by Land Salzburg under grant number 20204-WISS/263/6-6022.
Antonis Skarlatos: Antonis Skarlatos is funded by the European Union (ERC grant, DYNALP, 101170133). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

1 Introduction

Clustering is a fundamental problem in data analysis and classification. One of the most classical k-clustering problems is the k-center problem, which has been studied in various settings [15, 9, 2]. Given a metric space $(\mathcal{X}, \text{dist})$, a set of points $P \subseteq \mathcal{X}$ with $|P| = n$, and an integer $k \leq n$, the goal of the k-center problem is to output a subset $S \subseteq P$ of at most k points, called *centers*, such that the maximum distance from every point in P to its closest center is minimized. The k-center problem admits polynomial-time 2-approximation algorithms [16, 12], and it is known to be NP-hard to approximate the k-center objective within a factor of $(2 - \epsilon)$ for any constant $\epsilon > 0$ [17].

In the *fully dynamic* setting, the adversary can insert and delete points. In the *partially dynamic* setting, only one type of adversarial update – either insertions or deletions – is allowed. For example, in the *incremental* setting, the set of centers S is maintained under adversarial point insertions into P. The k-clustering problems in the dynamic setting with recourse have attracted a lot of attention [7, 19, 10, 11, 6, 5, 14].

© Mara Grilnberger and Antonis Skarlatos;
licensed under Creative Commons License CC-BY 4.0

Worst-case recourse

The *worst-case recourse* is the number of replacements made to the set of centers after any adversarial point update. In general, worst-case guarantees are crucial in real-time systems, and extensive research efforts have focused on achieving them [8, 3, 20, 4, 1, 14, 13]. From a theoretical perspective, minimizing worst-case recourse is a fundamental combinatorial problem, as it captures how smoothly solutions adapt in response to adversarial updates. From a practical standpoint, consider a scenario where the centers are used for an expensive task, and any modification to the solution introduces latency. Amortized recourse guarantees may still allow occasional long delays, whereas worst-case recourse guarantees ensure that no significant delays occur at any step.

To understand the limitations of recourse, an interesting direction is to study the best approximation ratio achievable by a polynomial-time algorithm with optimal recourse bounds. Fully dynamic algorithms with small recourse tend to have large approximation ratios. Hence, it is natural to study the incremental setting when the recourse is optimal – namely, a worst-case recourse of 1.

1.1 Upper Bounds

The area of research focusing on optimizing the recourse is commonly referred to as *consistent clustering*, a concept introduced by Lattanzi and Vassilvitskii [19]. In the incremental consistent setting, the "doubling algorithm" of Charikar, Chekuri, Feder, and Motwani [7] achieves an 8-approximation; their incremental algorithm maintains k clusters and only permits merging two clusters. The same work also presents a 6-approximation algorithm that requires solving the NP-hard clique partition problem as a subroutine. More recently, Forster and Skarlatos [11] obtained a 6-approximation algorithm in the incremental setting with a worst-case recourse of 1 and a worst-case update time of $O(k^2)$.[1] To compare their incremental algorithm with the "doubling algorithm", the authors in [11] showed that a slightly modified version of the latter [7] admits a worst-case recourse of 1 as well. In this direction, we improve upon prior work by reducing the approximation ratio to $3+\sqrt{5} \approx 5.23$.

▶ **Theorem 1.** *There is a deterministic incremental algorithm for the k-center clustering problem that, given a point set P in a metric space subject to point insertions and an integer $k \geq 1$, maintains a subset of points $S \subseteq P$ with $|S| = k$ such that:*

- *The set S is a $(3+\sqrt{5})$-approximate k-center solution.*
- *The worst-case recourse is at most 1.*
- *The worst-case update time is $O(k^2)$.*

Interestingly, our improved approximation ratio beats even the randomized algorithm of [7], which yields a $2e \approx 5.43$ approximation. As a result, our deterministic algorithm improves on a longstanding bound from more than 25 years ago. To obtain this improvement, motivated by the idea of [11] to combine the two standard static k-center algorithms – Gonzalez's algorithm and the Hochbaum–Shmoys algorithm [16] – we introduce new ideas and provide a more fine-grained and sophisticated analysis of the approximation ratio.

[1] The main result of [11] is, however, their fully dynamic algorithm.

1.2 Lower Bounds

Another interesting direction is to find the best approximation ratio for some fixed worst-case recourse bound. While there are results giving lower bounds on amortized recourse [19], namely a lower bound of $\Omega(\frac{k \log n}{n})$ changes, nothing is known about lower bounds for worst-case recourse. We initiate this line of work by proving the following theorem for the partially dynamic setting.

▶ **Theorem 2.** *Consider a point set P in a metric space subject to point insertions, an integer $k \geq 1$, and an approximation factor $\alpha \in [1, \sqrt{2})$. Then any partially dynamic α-approximate k-center algorithm has a worst-case recourse of k.*

Recently, Kozma [18] investigates the *nested* setting, where the k-center solution has to be built center by center, giving a good solution at each step (where the number of centers is at most k). While this differs from the incremental setting that we consider, two of the presented lower bounds transfer to "our" incremental setting, where we allow adversarial point insertions. This is because the adversary can add $k-1$ points to their examples that are at a distance "infinity" from all other points, leaving less centers for the original points at each insertion. Once the number of centers on the original points becomes small enough, by their arguments, the same bounds follow. As a result, the actual lower bound for an approximation factor is at least $2\cos\left(\frac{\pi}{7}\right)$ (see Section 2.2 in [18]).

2 Preliminaries

Consider a point set P in a metric space $(\mathcal{X}, \mathrm{dist})$ with $|P| = n$. We omit the notation $(\mathcal{X}, \mathrm{dist})$ when the context is clear. For each point $p \in P$ and each subset $S \subseteq P$, the *distance* from p to S is equal to $\mathrm{dist}(p, S) \coloneqq \min_{q \in S} \mathrm{dist}(p, q)$.

▶ **Definition 3** (k-center clustering problem)**.** *Given a point set P in a metric space and an integer $k \geq 1$, the goal of the k-center clustering problem is to output a subset $S \subseteq P$ of at most k points such that the value $\max_{p \in P} \mathrm{dist}(p, S)$ is minimized.*

The points in S are referred to as *centers*, and the (k-center) *cost* of S is equal to $\mathrm{cost}(S) \coloneqq \max_{p \in P} \mathrm{dist}(p, S)$. The *optimal* k-center cost is denoted by $R^* \coloneqq \min_{|S| \leq k} \mathrm{cost}(S)$. In the *improper* k-center clustering problem, the k-center solution S is allowed to include points from the metric space that are not contained in the point set P.

▶ **Definition 4** (improper k-center clustering problem)**.** *Given a point set P in a metric space $(\mathcal{X}, \mathrm{dist})$ and an integer $k \geq 1$, the goal of the improper k-center clustering problem is to output a subset $S \subseteq \mathcal{X}$ of at most k points such that the value $\max_{p \in P} \mathrm{dist}(p, S)$ is minimized.*

The improper k-center clustering problem is defined for the following technical reason. The optimal k-center cost $\hat{R}^*$ of the improper version is non-decreasing under adversarial point insertions, whereas the optimal k-center cost R^* of Definition 3 may decrease after an adversarial point insertion. Moreover, by definition it holds that $\hat{R}^* \leq R^*$. Therefore, we can upper bound the cost of our k-center solution with respect to the monotonic $\hat{R}^*$, which in turn is upper bounded by the desired optimal k-center cost R^*.

In the *incremental* setting, the set of centers S is maintained under adversarial point insertions into P. The *worst-case recourse* is the number of replacements made to the set of centers after any adversarial point insertion. Note that deterministic dynamic algorithms work against an adaptive adversary.

3 Technical Overview

3.1 Incremental Algorithm

Our incremental k-center algorithm is given an incremental point set P that is subject to adversarial point insertions. The algorithm unfolds in *phases*, where each phase can last for multiple adversarial point insertions. During a phase the radius r remains unchanged, and when the subsequent phase begins the radius r is increased by at least a factor of $\alpha > 1$.

Each center is labeled as either *leader* or *regular*. Each point is assigned to its responsible center, and the corresponding distance provides an upper bound on the cost of the k-center solution. A leader center is responsible for points within distance $\alpha r + \beta r$, while a regular center (at the beginning of a phase) is responsible for points within distance βr. Whenever the radius r is increased, some centers become regular. For the updated radius r, every point lies within distance βr from its nearest center. Therefore, the parameters α and β should satisfy $\alpha r + \beta r = \alpha\beta r$, which means that $\alpha + \beta = \alpha\beta$.

Gonzalez Operation

At the beginning of each phase, the centers are ordered using Gonzalez's algorithm [12], which iteratively selects the point farthest from the previously chosen centers as the next center. Any center within distance αr of another center with a lower index is labeled as regular, and the rest are labeled as leaders.

3.1.1 Approximation Ratio Within a Phase

At the last adversarial point insertion of a phase, all pairwise distances between centers are greater than $\alpha^i \cdot r$, all centers are leaders, and the newly inserted point is at a distance greater than $\alpha^i \cdot r$ from every center, where $i \geq 1$ is an integer. The radius r is then increased to at least $\alpha^i \cdot r$, which implies that $r \leq 2\hat{R}^* \leq 2R^*$ for the increased radius r. As new points are inserted by the adversary, the *primary goal* during a phase is to adjust the centers so that the minimum pairwise distance between them is at least αr. The leader centers remain fixed, while the regular centers are allowed to *move* in order to achieve the primary goal. Whenever a regular center c moves, it becomes a leader center and the incremental algorithm ensures that the distance between c and all other centers is greater than βr. Furthermore, the algorithm ensures that c is at a distance greater than αr from any other leader center. Once the primary goal is fulfilled, the phase ends and the subsequent phase begins.

Subphases

Notice that when a regular center moves, some points may lose their assigned center. In turn, another center c becomes responsible for these points, potentially increasing the distance from c to points assigned to it. This center c might also move as a result of another adversarial point insertion, causing a chain of changes in which some points are repeatedly reassigned to different centers. Since these adjustments affect the approximation ratio, for the purpose of analyzing the approximation ratio, we divide each phase into two subphases.

The first subphase is active while the pairwise distances of centers are at least r. The second subphase begins when the minimum distance between two centers becomes greater than βr. Since the incremental algorithm ensures that the distance between a center that moves and all other centers is greater than βr, the minimum distance between any two centers stays greater than βr until the end of the phase.

First subphase. During the first subphase, the pairwise distances of centers are at least r, which implies that $r \leq 2R^*$. Moreover during the first subphase, the minimum distance between two centers involves only centers that have not moved. In turn due to the Gonzalez Operation, the old position in the metric space of every *replaced* center is within distance $2R^*$ from its closest center. As a result, all points are within distance $\max(2R^* + \beta r, \alpha r)$ from their closest center.

Second subphase. During the second subphase, the pairwise distances of centers are greater than βr, which implies that $\beta r \leq 2R^*$. Since every regular center is within distance αr from a leader center, the old position in the metric space of every *replaced* center is within distance αr from its closest center. As a result, all points are within distance $\alpha r + \beta r$ from their nearest center. Observe that as $\beta r \leq 2R^*$, we have $\alpha r \leq 2R^* \cdot \frac{\alpha}{\beta}$. Consequently, it follows that all points are within distance $2R^* \cdot (\frac{\alpha}{\beta} + 1)$ from their closest center.
Finally, the values for the parameters α and β that satisfy the requirements (i.e., $1 \leq \beta < \alpha$ and $\alpha + \beta \leq \alpha\beta$) and minimize the approximation ratio are $\alpha = \frac{3+\sqrt{5}}{2}$ and $\beta = \frac{1+\sqrt{5}}{2}$. The resulting approximation ratio of the incremental algorithm is $3 + \sqrt{5}$.

4 Incremental Consistent k-Center Clustering

In the incremental setting, the point set P undergoes point insertions sent by the adaptive adversary. Our goal is to develop an incremental algorithm for the k-center clustering problem with small approximation ratio, low update time, and worst-case recourse of 1. In this section, we develop a deterministic incremental algorithm whose guarantees are demonstrated in Theorem 1. Recall that R^* denotes the current optimal k-center cost of the given instance.

▶ **Theorem 1.** *There is a deterministic incremental algorithm for the k-center clustering problem that, given a point set P in a metric space subject to point insertions and an integer $k \geq 1$, maintains a subset of points $S \subseteq P$ with $|S| = k$ such that:*

- *The set S is a $(3 + \sqrt{5})$-approximate k-center solution.*
- *The worst-case recourse is at most 1.*
- *The worst-case update time is $O(k^2)$.*

State of the incremental algorithm

Our incremental algorithm maintains an ordered set $S = \{c_1, \ldots, c_k\}$ of k centers, a *radius* r, and the minimum pairwise distance between centers $minDistS := \min_{x,y \in S,\, x \neq y} \mathrm{dist}(x, y)$. Each center is labeled as either *regular* or *leader*, and we denote by S_{reg} and S_{ldr} the sets of regular and leader centers respectively. Two parameters α and β are used in the algorithm satisfying $1 \leq \beta < \alpha$ and $\alpha + \beta \leq \alpha\beta$; the specific values of these parameters are $\alpha = \frac{3+\sqrt{5}}{2}$ and $\beta = \frac{1+\sqrt{5}}{2}$ (i.e., β is the golden ratio).

Invariants for the approximation ratio

The following four invariants are satisfied throughout the incremental algorithm. By enforcing Invariant 1 and Invariant 2, we deduce that $r \leq 2R^*$. Using Invariant 3 and a more fine-grained version of Invariant 4, we prove that the approximation ratio is $3 + \sqrt{5}$. These invariants are responsible solely for the approximation ratio, so our aim is to satisfy them with a worst-case recourse of 1.

Invariant 1: $|S| = k$.

Invariant 2: There is a point $p_S \in P \setminus S$ such that the pairwise distances of points in $S \cup \{p_S\}$ are at least r.

Invariant 3: For every regular center $c_1 \in S_{\text{reg}}$, there is a leader center $c_2 \in S_{\text{ldr}}$ such that $\text{dist}(c_1, c_2) \leq \alpha r$.

Invariant 4: Every point $p \in P$ is within distance $\alpha r + \beta r$ from its closest center in S.

4.1 Incremental Algorithm

We begin by describing the Gonzalez Operation, which is central to the incremental algorithm and induces an ordering of the centers. Next, we present the preprocessing phase of the incremental algorithm, and then explain how the algorithm handles adversarial point insertions. A pseudocode of the incremental algorithm is provided in Algorithm 1. Recall that α and β satisfy $1 \leq \beta < \alpha$ and $\alpha + \beta \leq \alpha\beta$.

Gonzalez Operation

During the Gonzalez Operation, the algorithm sets c_1 to be an arbitrary point from S and labels it as a leader. In each subsequent iteration $i \geq 2$, the algorithm sets c_i to be a point from $S \setminus \{c_1, \ldots, c_{i-1}\}$ that is farthest from the already assigned centers $\{c_1, \ldots, c_{i-1}\}$. If c_i is within distance αr from a leader, it is labeled as regular; otherwise, c_i is labeled as a leader. Observe that this procedure results in an index $j \in [1, k]$ such that $S_{\text{ldr}} = \{c_1, \ldots, c_j\}$ and $S_{\text{reg}} = \{c_{j+1}, \ldots, c_k\}$.

4.1.1 Preprocessing Phase

The preprocessing phase is similar to that of the incremental algorithm in [11]. It utilizes the following method by Hochbaum and Shmoys [16] that repeatedly computes a *distance-r independent set*.

▶ **Definition 5** (Distance-r Independent Set)**.** *Given a point set P in a metric space and a parameter $r > 0$, a distance-r independent set $S \subseteq P$ is a subset of points such that $\min_{p,q \in S} \text{dist}(p, q) > r$.*

▶ **Lemma 6** ([16])**.** *Consider a k-center instance (P, k), where P is a point set in a metric space, and let D be the sorted set of pairwise distances among points in P. By applying binary search on the set D and calling a maximal distance-r independent set algorithm $\mathcal{A}$ at each iteration, we can find a 2-approximate solution for the k-center instance. The running time of this algorithm is $O(\mathcal{T}_{\mathcal{A}} \cdot \log |P|)$, where $\mathcal{T}_{\mathcal{A}}$ is the running time of $\mathcal{A}$.*

In the preprocessing phase, we run the method of Lemma 6 to obtain a 2-approximate solution S. Let $r \coloneqq \max_{p \in P} \text{dist}(p, S)$ be the radius of S. The next task is to ensure that all four invariants are satisfied. To achieve this, the following steps are executed while $|S| < k$:

- If there is a point at a distance of at least r from S, then it is added to S.
- Otherwise, r is updated to $\frac{r}{\beta}$.

Once $|S| = k$, the radius r is repeatedly divided by β until there is a point $p \in P \setminus S$ such that $\text{dist}(p, S) \geq r$. Then the Gonzalez Operation is run, which assigns indices to the centers so that each subsequent center is the farthest from the previously selected centers. Additionally, the Gonzalez Operation assigns labels such that there is an index j for which all centers with index at most j are leaders, and all centers with index greater than j are regular.

Note that the point set P may initially contain at most k points. In this special case, the preprocessing phase continues while $|P| \leq k$, and all points in P become centers. Some invariants might be violated in this special case, but this is not an issue as every point is a center. Once $|P| = k + 1$, the computation of the set S and the radius r is performed as described before. Notice that the worst-case recourse is 1, since all points except one belong to S after the computation. Finally, the Gonzalez Operation is executed to determine the sets S_{reg} and S_{ldr}.

Analysis of the preprocessing phase

In this paragraph, we show that all four invariants are satisfied after the preprocessing phase. At the end of the preprocessing phase, we have $|S| = k$ and so Invariant 1 is satisfied. By the choice of S and r given by the method in Lemma 6 and of the preprocessing phase, there is a point $p_S \in P \setminus S$ such that the pairwise distances of points in $S \cup \{p_S\}$ are at least r, as required for Invariant 2. The validity of Invariant 3 follows by the construction of the Gonzalez Operation. Regarding Invariant 4, observe that the radius r is divided by β only when there is no point $p \in P \setminus S$ with $\text{dist}(p, S) \geq r$. Consequently, no point lies at a distance of at least βr from the k-center solution S. Since $1 \leq \beta < \alpha$, for all points $p \in P$ it holds that $\text{dist}(p, S) < \beta r \leq \alpha r + \beta r$. Therefore, once the preprocessing phase has ended, all four invariants hold. Furthermore, the set S is a 2-approximate k-center solution because we run the method of Lemma 6, and adding more points to S can only reduce its approximation ratio.

4.1.2 Adversarial Point Insertion

When the adversary inserts a new point p^+ into the point set P (i.e., $P \coloneqq P \cup \{p^+\}$), the incremental algorithm proceeds as follows. If the new point p^+ is within distance βr from a regular center or within distance αr from a leader center, then the algorithm does nothing. Otherwise, the new point p^+ is at distance greater than βr from all centers and at distance greater than αr from all leader centers. In this scenario, the incremental algorithm continues based on whether the minimum pairwise distance between centers is at most αr.

Case 1: If $\mathit{minDistS} \leq \alpha r$, let c_i be the center that satisfies the following three conditions:
- $\text{dist}(c_i, S \setminus \{c_i\}) = \mathit{minDistS}$.
- c_i is a regular center.
- i is the largest possible index in $S = \{c_1, \ldots, c_k\}$.

The algorithm then replaces c_i in the set of centers S with the new point p^+, and sets $c_i \coloneqq p^+$.[2] Hence, the new point p^+ becomes the i-th center of the ordered set S. Finally, the algorithm labels $c_i = p^+$ as a leader.

Case 2: Otherwise if $\mathit{minDistS} > \alpha r$, let j be the smallest power such that:

$$\min\left(\text{dist}(p^+, S), \mathit{minDistS}\right) \leq \alpha^j r.$$

The algorithm then increases the radius r to $\alpha^{j-1} r$ and invokes the previously described Gonzalez Operation. Once the Gonzalez Operation finishes, the incremental algorithm repeats the initial steps for p^+ (see Line 33 in Algorithm 1), and notice that the algorithm either does nothing or proceeds as in Case 1.

[2] Note that c_i is a variable referring to a point in P. Hence, when we say "replace c_i in S with p^+", we mean replace the value of c_i in S with p^+. Following that, the value of c_i is updated to p^+.

Algorithm 1 Incremental Consistent k-Center.

▷ Let $S = \{c_1, \ldots, c_k\}$ be the ordered set of k centers
▷ Let r be the radius
▷ Let $minDistS \coloneqq \min_{x,y \in S,\, x \neq y} \mathrm{dist}(x, y)$
▷ The algorithm has global access to all its variables and data structures

function GonzalezOp()
  Let c_1 be an arbitrary center in S
  $j \leftarrow 1$
  for $i \in \{2, \ldots, k\}$ **do**
    $c_i \leftarrow \mathrm{argmax}_{c \in S \setminus \{c_1, \ldots, c_{i-1}\}} \mathrm{dist}(c, \{c_1, \ldots, c_{i-1}\})$
    if $\mathrm{dist}(c_i, \{c_1, \ldots, c_{i-1}\}) > \alpha r$ **then**
      $j \leftarrow i$
    end if
  end for
  $S \leftarrow \{c_1, \ldots, c_k\}$ ▷ Recall that S is an ordered set
  $S_{\mathrm{ldr}} \leftarrow \{c_1, \ldots, c_j\}$
  $S_{\mathrm{reg}} \leftarrow \{c_{j+1}, \ldots, c_k\}$
end function

function InsertPoint(p^+)
  Insert p^+ into P (if $p^+ \notin P$)
  if $\mathrm{dist}(p^+, S_{\mathrm{reg}}) \leq \beta r$ **or** $\mathrm{dist}(p^+, S_{\mathrm{ldr}}) \leq \alpha r$ **then**
    exit from InsertPoint()
  end if
  if $minDistS \leq \alpha r$ **then** ▷ case C1
    Find $c_i \in S_{\mathrm{reg}}$ that minimizes $\mathrm{dist}(c_i, S \setminus \{c_i\})$ and i is the largest possible index in $S = \{c_1, \ldots, c_k\}$
    Remove c_i from S and S_{reg}
    Add p^+ to S and S_{ldr}
    $c_i \leftarrow p^+$
  else ▷ case C2
    Find the smallest power $j+1$ such that $\min\left(\mathrm{dist}(p^+, S), minDistS\right) \leq \alpha^{j+1} r$
    $r \leftarrow \alpha^j r$
    ▷ New phase begins
    GonzalezOp()
    InsertPoint(p^+)
  end if
end function

4.2 Analysis of the Incremental Algorithm

In this section, we prove Theorem 1 by analyzing the incremental Algorithm 1. We denote by $S^{(\mathrm{old})}$ and $r^{(\mathrm{old})}$ the set of centers and the radius before the adversarial point insertion; the set of centers and the radius after the incremental algorithm processes the adversarial point insertion are denoted by S and r. Hence throughout the analysis, we use (old) as a superscript for the states before the adversarial point insertion, and omit the (old) superscript for the states after the incremental algorithm processes the adversarial point insertion. Recall that α and β satisfy $1 \leq \beta < \alpha$ and $\alpha + \beta \leq \alpha\beta$.

A contiguous block of executions of `InsertPoint()` during which the radius remains unchanged is called a *phase* of Algorithm 1. A phase ends when, upon an adversarial point insertion, the radius increases (i.e., r is updated to at least $\alpha r^{(\mathrm{old})}$); the next phase then begins with the next invocation of `InsertPoint()`.

▶ **Definition 7** (phase of the incremental algorithm). *A new phase begins after the invocation of `InsertPoint()` whenever the radius r has increased (i.e., $r > r^{(old)}$).*

4.2.1 Proof of Invariant 3 and Auxiliary Results

▶ **Lemma 8.** *After an adversarial point insertion, for every regular center $c_1 \in S_{reg}$ there is a leader center $c_2 \in S_{ldr}$ such that* $\text{dist}(c_1, c_2) \leq \alpha r$ *(i.e., Invariant 3 is satisfied).*

Proof. Notice that immediately after an execution of the Gonzalez Operation, a center is regular if and only if it is within distance αr from a leader center. Hence, it suffices to analyze independently each sequence of adversarial point insertions between two consecutive executions of the Gonzalez Operation. The proof proceeds by induction on the adversarial point insertions within a single such sequence. The base case is the beginning of such a sequence, where the statement holds due to the Gonzalez Operation. Whenever a recursive call to `InsertPoint`() after the Gonzalez Operation changes the solution S, the same argument as in the induction step of Case 1 applies.

For the induction step, consider an adversarial point insertion p^+. Observe that the incremental Algorithm 1 could not have proceeded with Case 2, as otherwise the Gonzalez Operation would be invoked, signaling the end of the current sequence. During Case 1, a regular center becomes a leader (see Lines 25 and 26 in Algorithm 1). Hence by the induction hypothesis, for every regular center c_1 there is a leader center c_2 such that $\text{dist}(c_1, c_2) \leq \alpha r^{(\text{old})}$. Since in Case 1 the radius does not increase, we have $r = r^{(\text{old})}$, and thus the claim follows. ◀

The following lemma shows that during Case 2 of the incremental Algorithm 1 (in which the Gonzalez Operation is executed), a new phase begins (i.e., the radius increases).

▶ **Lemma 9.** *After an adversarial point insertion, a new phase begins if and only if the Gonzalez Operation is executed.*

Proof. By the construction of the incremental Algorithm 1, the Gonzalez Operation is invoked only during Case 2. Since during Case 1 the radius remains unchanged, it suffices to prove that during Case 2 a new phase always begins. Let p^+ be the new adversarially inserted point. By construction, the incremental algorithm proceeds with Case 2 if and only if $\text{dist}(p^+, S_{\text{reg}}^{(\text{old})}) > \beta r^{(\text{old})}$, $\text{dist}(p^+, S_{\text{ldr}}^{(\text{old})}) > \alpha r^{(\text{old})}$ and $minDistS^{(\text{old})} > \alpha r^{(\text{old})}$.

Because $minDistS^{(\text{old})} > \alpha r^{(\text{old})}$, no two centers are within distance $\alpha r^{(\text{old})}$. Based on Lemma 8 (Invariant 3), every regular center is within distance $\alpha r^{(\text{old})}$ from a leader center, and thus it holds that $S_{\text{reg}}^{(\text{old})} = \emptyset$. Since $S^{(\text{old})} = S_{\text{ldr}}^{(\text{old})} \cup S_{\text{reg}}^{(\text{old})}$, it follows that $\text{dist}(p^+, S^{(\text{old})}) = \text{dist}(p^+, S_{\text{ldr}}^{(\text{old})}) > \alpha r^{(\text{old})}$. Therefore, it holds that $\min\left(\text{dist}(p^+, S^{(\text{old})}), minDistS^{(\text{old})}\right) > \alpha r^{(\text{old})}$, which implies that the power $j+1$ in Line 29 of Algorithm 1 is at least 2. As a result, the radius increases by at least a factor of α in Line 30 (i.e., $r \geq \alpha r^{(\text{old})}$). Since $\alpha > 1$, a new phase begins according to Definition 7, and thus the claim follows. ◀

The next two observations establish the correctness of Line 24 in Algorithm 1, which involves identifying a proper regular center to be sacrificed.

▶ **Observation 10.** *After an adversarial point insertion, for every two leader centers $c_1, c_2 \in S_{ldr}$ it holds that* $\text{dist}(c_1, c_2) > \alpha r$.

Proof. Notice that immediately after an execution of the Gonzalez Operation, any two leader centers are at distance greater than αr. Hence, it suffices to analyze each phase independently according to Lemma 9. The proof proceeds by induction on the adversarial point insertions within a single phase. The base case is the beginning of the phase, where the Gonzalez Operation is executed, after which `InsertPoint`() is called recursively. Since the

statement holds immediately after the Gonzalez Operation, the argument for the recursive call of `InsertPoint`() is identical to the induction step, where $S^{(\text{old})}$ corresponds to the set of centers immediately after the Gonzalez Operation.

For the induction step, consider an adversarial point insertion p^+. Observe that the set of leader centers S_{ldr} changes only if $\text{dist}(p^+, S_{\text{ldr}}^{(\text{old})}) > \alpha r^{(\text{old})}$. Moreover, the incremental Algorithm 1 must have proceeded with Case 1, as otherwise the next phase begins according to Lemma 9. During Case 1, a center c becomes a leader (i.e., c is added to S_{ldr}) only in Line 26 where $c = p^+$, and also $r = r^{(\text{old})}$. Therefore, together with the induction hypothesis, any two leader centers are at distance greater than αr within a single phase. ◀

The next observation indicates that whenever the incremental Algorithm 1 proceeds with Case 1, a regular center is successfully found to be replaced in the set of centers. This implies the correctness of Line 24 in Algorithm 1.

▶ **Observation 11.** *During Case 1 of the incremental Algorithm 1, there is a regular center* $c_i \in S_{reg}^{(old)}$ *such that* $\text{dist}(c_i, S^{(old)} \setminus \{c_i\}) = \mathit{minDistS}^{(old)}$.

Proof. Suppose towards a contradiction that $\mathit{minDistS}^{(\text{old})} \leq \alpha r^{(\text{old})}$ but for all regular centers $c \in S_{\text{reg}}^{(\text{old})}$ we have $\text{dist}(c, S^{(\text{old})} \setminus \{c\}) > \mathit{minDistS}^{(\text{old})}$. Since $S_{\text{ldr}}^{(\text{old})}$ and $S_{\text{reg}}^{(\text{old})}$ partition $S^{(\text{old})}$, there must be two leader centers $c_1, c_2 \in S_{\text{ldr}}^{(\text{old})}$ such that $\text{dist}(c_1, c_2) = \mathit{minDistS}^{(\text{old})} \leq \alpha r^{(\text{old})}$. However, this yields a contradiction based on Observation 10, and thus the claim follows. ◀

4.2.2 Worst-Case Update Time

Observe that the incremental Algorithm 1 in Case 2 recursively calls `InsertPoint`(p^+). The next lemma shows that there is at most one recursive call.

▶ **Lemma 12.** *The recursion depth of* `InsertPoint`() *is at most two.*

Proof. When Line 33 in Algorithm 1 is executed, then either $\alpha r \geq \text{dist}(p^+, S^{(\text{old})})$ or $\alpha r \geq \mathit{minDistS}^{(\text{old})}$, where r is the updated radius. Observe that $S^{(\text{old})}$ is the set of centers after the Gonzalez Operation in Line 32 and $\mathit{minDistS}^{(\text{old})}$ is the corresponding minimum pairwise distance among centers in $S^{(\text{old})}$. Consider the two cases separately after the recursive call to `InsertPoint`() in Line 33:

- Assume that $\mathit{minDistS}^{(\text{old})} \leq \alpha r$. In this case, the incremental algorithm either does nothing due to Line 20, or it proceeds with Case 1, in which no further recursive calls to `InsertPoint`() occur.
- Otherwise, assume that $\mathit{minDistS}^{(\text{old})} > \alpha r$ and $\text{dist}(p^+, S^{(\text{old})}) \leq \alpha r$. Let $S_{\text{ldr}}^{(\text{interm})}$ denote the set of leader centers after the execution of the Gonzalez Operation with the updated radius r. By the construction of the Gonzalez Operation, it holds that $S_{\text{ldr}}^{(\text{interm})} = S^{(\text{old})}$. In turn, this implies that $\text{dist}(p^+, S_{\text{ldr}}^{(\text{interm})}) = \text{dist}(p^+, S^{(\text{old})}) \leq \alpha r$. Hence, the condition of the if-statement in Line 20 is true, and the incremental algorithm terminates.

Consequently, there is at most one recursive call to `InsertPoint`(), and so the recursion depth of `InsertPoint`() is at most two. ◀

Therefore, the update time is dominated by the Gonzalez Operation that runs Gonzalez's algorithm [12] on the set S, which is of size k by Lemma 14 (Invariant 1).

▶ **Corollary 13.** *The worst-case update time of the incremental Algorithm 1 is* $O(k^2)$.

4.2.3 Proofs of Invariant 1 and Invariant 2

▶ **Lemma 14.** *After an adversarial point insertion, the size of S is k (i.e., Invariant 1 is satisfied). Moreover, the worst-case recourse is* 1.

Proof. In the preprocessing phase, the incremental algorithm constructs a solution S of k centers. During an adversarial point insertion, the set of centers S is altered only when the algorithm proceeds with Case 1. In this case, exactly one point is removed from S and the new adversarially inserted point is added to S (see Lines 25 and 26 in Algorithm 1). As a consequence, we have $|S| = k$ after every adversarial point insertion.

Notice that Case 1 occurs at most once per adversarial point insertion (see also Lemma 12). Thus, the set of centers changes by at most one point per adversarial update. ◀

The following lemma can be credited to Hochbaum and Shmoys [16], who presented a reduction from the k-center clustering problem to a variant of the maximal independent set problem. We denote by R^* the optimal k-center cost of Definition 3 and by $\hat{R}^*$ the optimal improper k-center cost of Definition 4.

▶ **Lemma 15** ([16])**.** *Consider a k-center instance (P, k), and let $S \subseteq P$ be a subset of points with $|S| \geq k+1$ such that for every pair of distinct points $p, q \in S$, we have $\mathrm{dist}(p, q) \geq r$. Then it holds that $r \leq 2\hat{R}^*$.*

Proof. Let S^* be an optimal improper k-center solution. Since the size of S^* is at most k, by the pigeonhole principle there exists a center $c^* \in S^*$ and two distinct points $p, q \in S$ such that $\mathrm{dist}(p, c^*) \leq \hat{R}^*$ and $\mathrm{dist}(q, c^*) \leq \hat{R}^*$. By the triangle inequality, we have $\mathrm{dist}(p, c^*) + \mathrm{dist}(q, c^*) \geq \mathrm{dist}(p, q) \geq r$. Hence, without loss of generality we can assume that $\mathrm{dist}(p, c^*) \geq \frac{r}{2}$. Since $\mathrm{dist}(p, c^*) \leq \hat{R}^*$, it follows that $\hat{R}^* \geq \frac{r}{2}$, as needed. ◀

▶ **Observation 16.** *Consider a k-center instance (P, k). Then it holds that $\hat{R}^* \leq R^*$.*

Proof. Observe that every k-center solution $S \subseteq P$ with $|S| \leq k$ is also a feasible solution with the same cost for the improper k-center clustering problem. Hence, comparing the respective optimal k-center solutions, it follows that $\hat{R}^* \leq R^*$. ◀

▶ **Lemma 17.** *After an adversarial point insertion, there is a point $p_S \in P \setminus S$ such that the pairwise distances of points in $S \cup \{p_S\}$ are at least r (i.e., Invariant 2 is satisfied). Moreover, the value of the radius r is at most $2\hat{R}^* \leq 2R^*$.*

Proof. The proof is by induction on the adversarial point insertions. The base case is immediately after the preprocessing phase, where Invariant 2 is satisfied. For the induction step, consider an adversarial point insertion p^+, and let P be the updated point set including p^+. If $\mathrm{dist}(p^+, S_{\mathrm{reg}}^{(\mathrm{old})}) \leq \beta r$ or $\mathrm{dist}(p^+, S_{\mathrm{ldr}}^{(\mathrm{old})}) \leq \alpha r$, then neither the set of centers nor the radius is modified (i.e., $S = S^{(\mathrm{old})}$ and $r = r^{(\mathrm{old})}$). By the induction hypothesis, there is a point $p_S \in (P \setminus \{p^+\}) \setminus S^{(\mathrm{old})}$ such that the pairwise distances of points in $S^{(\mathrm{old})} \cup \{p_S\}$ are at least $r^{(\mathrm{old})}$. Since $S = S^{(\mathrm{old})}$ and $r = r^{(\mathrm{old})}$ in this scenario, there is a point $p_S \in P \setminus S$ such that the pairwise distances of points in $S \cup \{p_S\}$ are at least r, as needed. Otherwise the incremental algorithm proceeds either with Case 1 or Case 2:

- Assume that the incremental algorithm proceeds with Case 1. In this case, it holds that $\mathrm{dist}(p^+, S^{(\mathrm{old})}) > \beta r^{(\mathrm{old})} \geq r^{(\mathrm{old})}$, as $\beta \geq 1$. By the construction of the algorithm, we have $S = (S^{(\mathrm{old})} \setminus \{c\}) \cup \{p^+\}$ for some $c \in S^{(\mathrm{old})}$ and the radius remains unchanged (i.e., $r = r^{(\mathrm{old})}$). By the induction hypothesis, the pairwise distances of points in $S^{(\mathrm{old})}$ are at least $r^{(\mathrm{old})} = r$. Thus by setting $p_S \coloneqq c$ and since $\mathrm{dist}(p^+, S^{(\mathrm{old})}) > r$, we infer that there is a point $p_S \in P \setminus S$ such that the pairwise distances of points in $S \cup \{p_S\}$ are at least r, as required.

- Assume that the incremental algorithm proceeds with Case 2. In this case, the radius r is updated to $\alpha^j r^{(\text{old})}$, where j is the largest power such that $\mathit{minDistS}^{(\text{old})} > \alpha^j r^{(\text{old})}$ and $\text{dist}(p^+, S^{(\text{old})}) > \alpha^j r^{(\text{old})}$. Since $S = S^{(\text{old})}$ at that moment, Invariant 2 holds after the radius r is assigned in Line 30 of Algorithm 1. Next, the incremental algorithm continues recursively with `InsertPoint`(p^+), where $r = \alpha^j r^{(\text{old})}$. Let $S_{\text{reg}}^{(\text{interm})}$ and $S_{\text{ldr}}^{(\text{interm})}$ denote the sets of regular and leader centers immediately after the execution of the Gonzalez Operation.
  - If $\text{dist}(p^+, S_{\text{reg}}^{(\text{interm})}) \leq \beta r$ or $\text{dist}(p^+, S_{\text{ldr}}^{(\text{interm})}) \leq \alpha r$, then the set of centers does not change (i.e., $S = S^{(\text{old})}$) and so Invariant 2 remains satisfied.
  - Otherwise, we have $S = (S^{(\text{old})} \setminus \{c\}) \cup \{p^+\}$ for some $c \in S^{(\text{old})}$. Thus by setting again $p_S \coloneqq c$ and since $\text{dist}(p^+, S^{(\text{old})}) > r$, we deduce that Invariant 2 remains satisfied. In other words, there is a point $p_S \in P \setminus S$ such that the pairwise distances of points in $S \cup \{p_S\}$ are at least r, as required.

Upper bound on r

By construction, the maintained k-center solution S is a subset of the point set P. Since Invariant 2 and Invariant 1 (see Lemma 14) are satisfied, there are $k+1$ points in P whose pairwise distances are at least r. Thus by Lemma 15 and Observation 16, it follows that $r \leq 2\hat{R}^* \leq 2R^*$. ◀

4.2.4 Approximation Ratio

In this subsection, we analyze the approximation ratio of our incremental algorithm, using a more fine-grained version of Invariant 4. The set S_{init} denotes the ordered set of centers at the beginning of the current phase; according to Lemma 9, the set S_{init} is the result after the last execution of the Gonzalez Operation. Observe that the set S_{init} may differ from the current set of centers $S^{(\text{old})}$ just before the incremental algorithm processes an adversarial point insertion. We note that S_{init} is an auxiliary set used in the following lemma.

▶ **Lemma 18.** *After an adversarial point insertion, for every point $p \in P$ it holds that either*

$$\text{dist}(p, S_{init}) \leq \beta r \quad \textit{or} \quad \text{dist}(p, S_{ldr}) \leq \alpha r.$$

Proof. The proof is by induction on the adversarial points insertions. The base case is immediately after the preprocessing phase, where $S_{\text{init}} = S$ and every point $p \in P$ is within distance βr from a center. For the induction step, consider an adversarial point insertion p^+, and let P be the updated point set including p^+. Since $S_{\text{reg}} \subseteq S_{\text{init}}$ at the beginning of each phase, and no new centers are added to S_{reg} during a phase, it follows that $S_{\text{reg}} \subseteq S_{\text{init}}$ throughout the algorithm.

In the scenario where $\text{dist}(p^+, S_{\text{reg}}^{(\text{old})}) \leq \beta r^{(\text{old})}$ or $\text{dist}(p^+, S_{\text{ldr}}^{(\text{old})}) \leq \alpha r^{(\text{old})}$, it holds that $\text{dist}(p^+, S_{\text{init}}) \leq \text{dist}(p^+, S_{\text{reg}}^{(\text{old})})$, $S_{\text{init}} = S_{\text{init}}^{(\text{old})}$, $r = r^{(\text{old})}$, $S_{\text{reg}} = S_{\text{reg}}^{(\text{old})}$, and $S_{\text{ldr}} = S_{\text{ldr}}^{(\text{old})}$, and so together with the induction hypothesis the correctness of the statement follows. Otherwise, the incremental algorithm proceeds either with Case 1 or Case 2:

- Assume that $\mathit{minDistS}^{(\text{old})} \leq \alpha r^{(\text{old})}$ (Case 1). Let $c_i \in S_{\text{reg}}^{(\text{old})}$ and $c_j \in S^{(\text{old})}$ such that $\text{dist}(c_i, c_j) = \mathit{minDistS}^{(\text{old})}$ and i is the largest possible index. Based on Observation 11, such centers c_i and c_j must exist. The algorithm replaces c_i in the set of centers with p^+ and labels p^+ as a leader. Since $S_{\text{reg}}^{(\text{old})} \subseteq S_{\text{init}}$, the old regular center c_i belongs to S_{init}. Hence, as in Case 1 we have $r = r^{(\text{old})}$ and $S_{\text{init}} = S_{\text{init}}^{(\text{old})}$, for all points in $P \setminus \{p^+\}$ the claim holds by the induction hypothesis. Additionally, for the inserted point p^+ we have $\text{dist}(p^+, S_{\text{ldr}}) = 0 \leq \alpha r$, as needed.

- Assume that $\mathit{minDistS}^{(\text{old})} > \alpha r^{(\text{old})}$ (Case 2); the incremental algorithm continues recursively with $\texttt{InsertPoint}(p^+)$ in Line 33. According to Lemma 9, a new phases begins which means that $r \geq \alpha r^{(\text{old})}$. For the updated radius r, the inserted point p^+ becomes a center if and only if (i) $\text{dist}(p^+, S_{\text{reg}}^{(\text{old})}) > \beta r$, (ii) $\text{dist}(p^+, S_{\text{ldr}}^{(\text{old})}) > \alpha r$, and (iii) $\mathit{minDistS}^{(\text{old})} \leq \alpha r$. First, we consider the newly inserted point p^+. If p^+ becomes a center during Case 1, then the claim holds for p^+. Otherwise based on Lemma 12, the algorithm does not continue with Case 2 again. Consequently, it holds that either $\text{dist}(p^+, S_{\text{reg}}) \leq \beta r$ or $\text{dist}(p^+, S_{\text{ldr}}) \leq \alpha r$, and since $S_{\text{reg}} \subseteq S_{\text{init}}$ the claim follows for p^+. It remains to consider the remaining points in $P \setminus \{p^+\}$. Since a new phase begins during Case 2, we have $r \geq \alpha r^{(\text{old})}$ and the set S_{init} changes. Let $S_{\text{init}}^{(\text{new})}$ be the set of centers after the Gonzalez Operation is invoked (i.e., $S_{\text{init}}^{(\text{new})} = S^{(\text{old})}$). Based on Lemma 8 (Invariant 3) and the fact that $\mathit{minDistS}^{(\text{old})} > \alpha r^{(\text{old})}$, we can infer that $S_{\text{ldr}}^{(\text{old})} = S^{(\text{old})}$, which means that $S_{\text{ldr}}^{(\text{old})} = S^{(\text{old})} = S_{\text{init}}^{(\text{new})}$. For an arbitrary point $q \in P \setminus \{p^+\}$, by the induction hypothesis either $\text{dist}(q, S_{\text{init}}^{(\text{old})}) \leq \beta r^{(\text{old})}$ or $\text{dist}(q, S_{\text{ldr}}^{(\text{old})}) \leq \alpha r^{(\text{old})}$; we analyze the two cases separately:
  - Consider a point $q \in P \setminus \{p^+\}$ such that $\text{dist}(q, S^{(\text{old})}) \leq \alpha r^{(\text{old})}$. Then as $\beta \geq 1$, it holds that:

    $$\text{dist}(q, S_{\text{init}}^{(\text{new})}) \leq \alpha r^{(\text{old})} \leq r \leq \beta r.$$

  - Consider a point $q \in P \setminus \{p^+\}$ such that $\text{dist}(q, S_{\text{init}}^{(\text{old})}) \leq \beta r^{(\text{old})}$ and $\text{dist}(q, S^{(\text{old})}) > \alpha r^{(\text{old})}$. Observe that leader centers are never removed from the set of centers during a (fixed) phase and that the distance from q to the set of centers increased during the (previous) phase. Hence, the closest center $c_q \in S_{\text{init}}^{(\text{old})}$ to q that was removed from the set of centers was a regular center (when the radius was $r^{(\text{old})}$). By Lemma 8 (Invariant 3), there must be another leader center $c \in S_{\text{ldr}}^{(\text{old})}$ such that $\text{dist}(c_q, c) \leq \alpha r^{(\text{old})}$. By the triangle inequality and since $\alpha + \beta \leq \alpha\beta$, we obtain that:

    $$\text{dist}(q, S_{\text{init}}^{(\text{new})}) \leq \alpha r^{(\text{old})} + \beta r^{(\text{old})} \leq \alpha\beta r^{(\text{old})} \leq \beta r.$$

  Since the algorithm is currently in Case 2, based on Lemma 12, the next recursive call of $\texttt{InsertPoint}(p^+)$ does not enter Case 2 again. This implies that the radius is no longer modified by this adversarial point insertion and the set S_{init} remains unchanged. If the solution changes due to Case 1 of the recursive $\texttt{InsertPoint}(p^+)$, the claim follows using the same arguments as before. ◀

To analyze the approximation ratio, we split every phase into two *subphases* and analyze each subphase separately; the subphases are defined as follows.

▶ **Definition 19** (subphases of a phase)**.** *During a phase of the incremental algorithm, the first subphase ends when the minimum pairwise distance between centers is greater than βr; the second subphase then begins.*

▶ **Observation 20.** *During a phase of the incremental algorithm, any $c \in S \setminus S_{init}$ that becomes a center after the beginning of the phase is at a distance greater than βr from all other centers until the end of the phase.*

Proof. According to Lemma 9, whenever the Gonzalez Operation is invoked during Case 2 of the incremental Algorithm 1, a new phase begins. Hence, it suffices to analyze only Case 1 of the incremental Algorithm 1. The proof is by induction on the adversarial point insertions within a single phase. The base case is at the beginning of the phase, where $S = S_{\text{init}}$ and

the statement trivially holds. For the induction step, consider an adversarial point insertion p^+. The set of centers is modified only in Case 1, which occurs only when the if-statement in Line 20 is false. This implies that for the new center p^+ (see Line 26 in Algorithm 1), it holds that $\text{dist}(p^+, S_{\text{reg}}^{(\text{old})}) > \beta r$ and $\text{dist}(p^+, S_{\text{ldr}}^{(\text{old})}) > \alpha r$, since $r = r^{(\text{old})}$ during a phase. In turn, it holds that $\text{dist}(p^+, S^{(\text{old})}) = \text{dist}(p^+, S_{\text{reg}}^{(\text{old})} \cup S_{\text{ldr}}^{(\text{old})}) > \beta r$. For any other center $c \in S^{(\text{old})} \setminus S_{\text{init}}$, the statement holds by the induction hypothesis, as $r = r^{(\text{old})}$. ◀

Hence, during the first subphase, the value *minDistS* is defined by two centers in $S \cap S_{\text{init}}$. From the start of the second subphase onward, *minDistS* exceeds βr until the end of the phase. This is because Observation 20 applies for any new center that is added until the end of the phase. Let $S_i \coloneqq \{c_j \in S_{\text{init}} \mid j \leq i\}$ denote the ordered subset of S_{init} up to center c_i, for any $1 \leq i \leq k$.

▶ **Lemma 21.** *Consider a fixed phase of the incremental algorithm. After an adversarial point insertion during the first subphase, assume that the incremental algorithm removes the i-th center c_i from the set of centers. Then the distance* $\text{dist}(c_i, S_{i-1})$ *equals the minimum pairwise distance between centers at that moment.*

Proof. Based on Observation 20 and Definition 19, the minimum pairwise distance between centers is attained between two centers in S_{init}. By the construction of the Gonzalez Operation, all leader centers have smaller index than regular centers in S_{init}. Since i is the largest such index in Line 24 of Algorithm 1 (during Case 1), it holds that $\text{dist}(c_i, S_{i-1}) \leq \text{dist}(c_i, S^{(\text{old})} \setminus \{c_i\})$, where c_i refers to the old location determined in Line 24. Based on Observation 11, we have $\text{dist}(c_i, S^{(\text{old})} \setminus \{c_i\}) = \mathit{minDistS}^{(\text{old})}$. Thus at the moment that c_i is removed from the set of centers, the minimum pairwise distance between centers is attained between c_i and S_{i-1}. ◀

▶ **Lemma 22.** *Consider a fixed phase of the incremental algorithm. After an adversarial point insertion during the first subphase, assume that the incremental algorithm removes the i-th center c_i from the set of centers. Then it holds that $S_{i-1} \subseteq S^{(old)}$ and $S_{i-1} \subseteq S$.*

Proof. After an adversarial insertion of a point p^+, the center c_i is replaced by p^+ in the set of centers (i.e., $S = (S^{(\text{old})} \setminus \{c_i\}) \cup \{p^+\}$). Since $p^+ \notin S_{i-1}$, it suffices to show that $S_{i-1} \subseteq S$. Suppose to the contrary that there is an adversarial point insertion such that c_i (referring to the old location) is removed from the set of centers and $S_{i-1} \not\subseteq S$, and consider the first such adversarial point insertion. Let c_j be the center (referring to the old location) with the smallest index j in S_{init} which has been removed from the set of centers in a previous adversarial point insertion (i.e., $c_j \in S_i \setminus S$ with the smallest possible index j). By the minimality of the counter-example, it holds that $S_{j-1} \subseteq S$ and $j < i$, which implies that c_i does not belong to S_{j-1}. By the construction of the Gonzalez Operation, we have $\text{dist}(c_i, S_{j-1}) \leq \text{dist}(c_j, S_{j-1})$.

Let τ be the moment when c_j was removed from the set of centers. Since the phase is fixed, the τ-th adversarial point insertion occurs during the same first subphase. Based on Lemma 21, the minimum pairwise distance in Line 24 of Algorithm 1 (during Case 1) at time τ is achieved between c_j and S_{j-1}. However, as $\text{dist}(c_i, S_{j-1}) \leq \text{dist}(c_j, S_{j-1})$ at the τ-th adversarial point insertion, the minimum pairwise distance between centers at time τ could also be achieved with c_i. Since $i > j$, the incremental algorithm should have replaced c_i before replacing c_j, which leads to a contradiction. ◀

Recall that the optimal k-center cost of Definition 3 is denoted by R^*, and the optimal improper k-center cost of Definition 4 is denoted by $\hat{R}^*$.

▶ **Observation 23.** *Consider a k-center instance (P, k) subject to adversarial point insertions. Then $\hat{R}^*$ is non-decreasing at all times.*

Proof. Consider an optimal improper k-center solution $\hat{S}_1^*$ before the adversarial point insertion, where $\text{cost}^{(\text{old})}(\hat{S}_1^*) = (\hat{R}^*)^{(\text{old})}$. Suppose to the contrary that after an adversarial point insertion, the optimal improper k-center cost decreases (i.e., $\hat{R}^* < (\hat{R}^*)^{(\text{old})}$). Under adversarial point insertions, the cost of $\hat{S}_1^*$ is non-decreasing (i.e., $\text{cost}^{(\text{old})}(\hat{S}_1^*) \leq \text{cost}(\hat{S}_1^*)$). Thus, there must be another optimal k-center solution $\hat{S}_2^*$ after the adversarial point insertion, where $\text{cost}(\hat{S}_2^*) = \hat{R}^*$.

Since $\hat{S}_2^*$ is a subset of the metric space and $|\hat{S}_2^*| \leq k$, the set $\hat{S}_2^*$ is also a feasible improper k-center solution before the adversarial point insertion. The cost of $\hat{S}_2^*$ is non-decreasing under adversarial point insertions, and so $\text{cost}^{(\text{old})}(\hat{S}_2^*) \leq \text{cost}(\hat{S}_2^*)$. Combining all these together, it follows that $\text{cost}^{(\text{old})}(\hat{S}_2^*) \leq \text{cost}(\hat{S}_2^*) = \hat{R}^* < (\hat{R}^*)^{(\text{old})}$, contradicting the fact that $\hat{S}_1^*$ was an optimal improper k-center solution before the adversarial point insertion. Therefore, the optimal improper k-center cost cannot decrease when a new point is inserted by the adversary. ◀

▶ **Lemma 24.** *Let $\alpha := \frac{3+\sqrt{5}}{2}$ and $\beta := \frac{1+\sqrt{5}}{2}$. After an adversarial point insertion, for every point $p \in P$ it holds that:*

$$\text{dist}(p, S) \;\leq\; (3+\sqrt{5})\hat{R}^* \;\leq\; (3+\sqrt{5})R^*.$$

Proof. Based on Lemma 18, every point $p \in P$ is either within distance βr from the set S_{init} or within distance αr from a leader center in S_{ldr}. In the latter case, by Lemma 17 we have $r \leq 2\hat{R}^* \leq 2R^*$, and as $S_{\text{ldr}} \subseteq S$ it follows that $\text{dist}(p, S) \leq \alpha r \leq 2\alpha\hat{R}^* = (3+\sqrt{5})\hat{R}^* \leq (3+\sqrt{5})R^*$, as needed. Since $\beta < \alpha$, the same conclusion holds for any point $p \in P$ satisfying $\text{dist}(p, c) \leq \beta r$ for some center $c \in S_{\text{init}} \cap S_{\text{reg}}$. Notice that when a new phase begins then $S_{\text{init}} = S_{\text{reg}} \cup S_{\text{ldr}}$, implying the correctness of the statement by Lemma 18 together with the fact that $r \leq 2\hat{R}^* \leq 2R^*$. Hence, it remains to analyze points $p \in P$ such that $\text{dist}(p, S_{\text{ldr}}) > \alpha r$ and $\text{dist}(p, S_{\text{init}}) = \text{dist}(p, q) \leq \beta r$ for some $q \in S_{\text{init}} \setminus S$ during a fixed phase. Namely, the point q has been replaced as a center during this phase.

The proof is by induction on the adversarial point insertions within a single phase. The base case is at the beginning of the phase, where $S_{\text{init}} = S$ and the statement holds as explained earlier. For the induction step, consider an adversarial point insertion p^+, and let P be the updated point set including p^+. In the scenario where $\text{dist}(p^+, S_{\text{reg}}^{(\text{old})}) \leq \beta r^{(\text{old})}$ or $\text{dist}(p^+, S_{\text{ldr}}^{(\text{old})}) \leq \alpha r^{(\text{old})}$, neither the set of centers nor the radius is modified (i.e., $S = S^{(\text{old})}$ and $r = r^{(\text{old})}$). Thus the statement holds by the induction hypothesis combined with Observation 23 and Observation 16. Otherwise it holds that $\text{dist}(p^+, S^{(\text{old})}) > \beta r^{(\text{old})}$, and the incremental algorithm continues either with Case 1 or Case 2. According to Lemma 9, during Case 2 of the incremental Algorithm 1, a new phase begins. Therefore, we only need to consider adversarial point insertions for which the incremental algorithm proceeds with Case 1 in the current phase.

For clarity regarding the reset of c_i in Line 27 of Algorithm 1, let $c_i^{(\text{old})}$ denote the old center (referring to the old location) determined in Line 24. Since the radius does not increase in Case 1, we have $r = r^{(\text{old})}$. Observe that the newly inserted point p^+ becomes a leader in Line 26 of Algorithm 1, and by construction we have $S_{\text{reg}} = S_{\text{reg}}^{(\text{old})} \setminus \{c_i^{(\text{old})}\}$ and $S_{\text{ldr}} = S_{\text{ldr}}^{(\text{old})} \cup \{p^+\}$ (i.e., $S = (S^{(\text{old})} \setminus \{c_i^{(\text{old})}\}) \cup \{p^+\}$). This implies that for all points whose nearest center in $S^{(\text{old})}$ is not $c_i^{(\text{old})}$, the claim follows by the induction hypothesis combined with Observation 23 and Observation 16. Hence, it suffices to consider points $p \in P^{(\text{old})}$ whose nearest center in $S^{(\text{old})}$ is the point $c_i^{(\text{old})}$. For a fixed such point $p \in P^{(\text{old})}$,

let $q \in S_{\text{init}}$ be its nearest center at the beginning of the current phase. Recall that for a point $p \in P$, its closest point q in S_{init} may not belong to the set of centers before or after the adversarial update (i.e., S_{init} can differ from $S^{(\text{old})}$ and S). There are two cases to analyze depending on which subphase of the incremental Algorithm 1 is active:

The first subphase is active. Let i be the index determined in Line 24 of Algorithm 1. Since the first subphase is active (see Definition 19), it holds that $\text{dist}(c_i^{(\text{old})}, S^{(\text{old})} \setminus \{c_i^{(\text{old})}\}) = \mathit{minDistS}^{(\text{old})} \leq \beta r$, which implies that $\text{dist}(c_i^{(\text{old})}, S^{(\text{old})} \setminus \{c_i^{(\text{old})}\}) < \text{dist}(p^+, S^{(\text{old})})$. Since the distance between $c_i^{(\text{old})}$ and $S^{(\text{old})} \setminus \{c_i^{(\text{old})}\}$ is the minimum over all pairwise distances of points in $S^{(\text{old})}$, the pairwise distances of points in $S^{(\text{old})} \cup \{p^+\}$ are at least $\text{dist}(c_i^{(\text{old})}, S^{(\text{old})} \setminus \{c_i^{(\text{old})}\})$. Based on Lemma 14 we have $|S^{(\text{old})} \cup \{p^+\}| = k+1$, and observe that $S^{(\text{old})} \cup \{p^+\} \subseteq P$. As a consequence, by Lemma 15 it holds that $\text{dist}(c_i^{(\text{old})}, S^{(\text{old})} \setminus \{c_i^{(\text{old})}\}) \leq 2\hat{R}^*$.

Observe that if $q = c_i^{(\text{old})}$ then by the triangle inequality and by Observation 16, we have $\text{dist}(p, S) \leq \beta r + \text{dist}(q, S) \leq \beta r + 2\hat{R}^* \leq (1+\beta)2\hat{R}^* = (3+\sqrt{5})\hat{R}^* \leq (3+\sqrt{5})R^*$, as needed for p. Otherwise $q \notin S^{(\text{old})}$ and based on Lemma 22, we can infer that $q \notin S_{i-1}$. Due to the order induced by the Gonzalez Operation, we have $\text{dist}(q, S_{i-1}) \leq \text{dist}(c_i^{(\text{old})}, S_{i-1})$. Again by Lemma 22, it follows that $\text{dist}(q, S) \leq \text{dist}(q, S_{i-1})$, and so $\text{dist}(q, S) \leq \text{dist}(c_i^{(\text{old})}, S_{i-1})$. Based on Lemma 21 (note that i is the largest such index in Line 24), it holds that $\text{dist}(c_i^{(\text{old})}, S_{i-1}) \leq \text{dist}(c_i^{(\text{old})}, S^{(\text{old})} \setminus \{c_i^{(\text{old})}\})$. Combining these inequalities, we conclude that $\text{dist}(q, S) \leq \text{dist}(c_i^{(\text{old})}, S^{(\text{old})} \setminus \{c_i^{(\text{old})}\})$. Since we previously showed that $\text{dist}(c_i^{(\text{old})}, S^{(\text{old})} \setminus \{c_i^{(\text{old})}\}) \leq 2\hat{R}^*$, we deduce that $\text{dist}(q, S) \leq 2\hat{R}^*$. As a result, by the triangle inequality it follows that $\text{dist}(p, S) \leq 2\hat{R}^* + \beta r$. By Lemma 17 we have $r \leq 2\hat{R}^* \leq 2R^*$, and thus $\text{dist}(p, S) \leq (3+\sqrt{5})\hat{R}^* \leq (3+\sqrt{5})R^*$, as required for p.

The second subphase is active. At any moment during the second subphase, the minimum pairwise distance between centers is greater than $\beta r^{(\text{old})}$ (i.e., $\mathit{minDistS}^{(\text{old})} > \beta r^{(\text{old})}$). As $r = r^{(\text{old})}$ we have $\text{dist}(p^+, S^{(\text{old})}) > \beta r$, which means that the pairwise distances of points in $S^{(\text{old})} \cup \{p^+\}$ are at least βr. According to Lemma 14 we have $|S^{(\text{old})} \cup \{p^+\}| = k+1$, and thus by Lemma 15 it holds that $\beta r \leq 2\hat{R}^*$. Note that the point q belongs to S_{init} but is no longer a center (i.e., q was already replaced by another point in the set of centers during this phase). By the construction of the incremental Algorithm 1, only regular centers are removed from the set of centers. In turn, according to Lemma 8 (Invariant 3) and since the radius remains unchanged during a fixed phase, there is a leader center within distance αr from the point q. Therefore, by the triangle inequality and as $r \leq \frac{2\hat{R}^*}{\beta}$ and $\hat{R}^* \leq R^*$, it follows that $\text{dist}(p, S) \leq \alpha r + \beta r \leq \frac{\alpha}{\beta} 2\hat{R}^* + 2\hat{R}^* = (\frac{\alpha}{\beta} + 1)2\hat{R}^* = (3+\sqrt{5})\hat{R}^* \leq (3+\sqrt{5})R^*$, as required for p. ◀

Proof of Theorem 1. The proof follows by Lemmas 14 and 24 and Corollary 13. ◀

5 Lower Bound on Worst-Case Recourse

In this section, we show a lower bound on the worst-case recourse. In the following, we give the proof for the incremental setting. For the decremental setting, this proof can be "reversed", starting with p^+ as part of the original point set. Then, the adversary deletes that point and the original setting of the following proof needs to be restored, requiring k centers to be moved.

▶ **Theorem 2.** *Consider a point set P in a metric space subject to point insertions, an integer $k \geq 1$, and an approximation factor $\alpha \in [1, \sqrt{2})$. Then any partially dynamic α-approximate k-center algorithm has a worst-case recourse of k.*

Proof. Consider the following k-center instance containing $3k-2$ points $p_1, p_2, \ldots, p_{3k-2}$ such that:

- $\text{dist}(p_1, p_2) = \text{dist}(p_2, p_3) = d'$,
- $\text{dist}(p_{i-1}, p_i) = 2d'$, for all $i \geq 4$ with $i \equiv 1 \pmod 3$,
- $\text{dist}(p_{i-1}, p_i) = d$, for all $i \geq 5$ with $i \not\equiv 1 \pmod 3$, and
- $\text{dist}(p_1, p_i) = \text{dist}(p_1, p_{i-1}) + \text{dist}(p_{i-1}, p_i)$ for all i,

where d and d' satisfy $\max(\frac{2}{3}, \frac{\alpha}{2})d < d' < \frac{d}{\alpha}$.

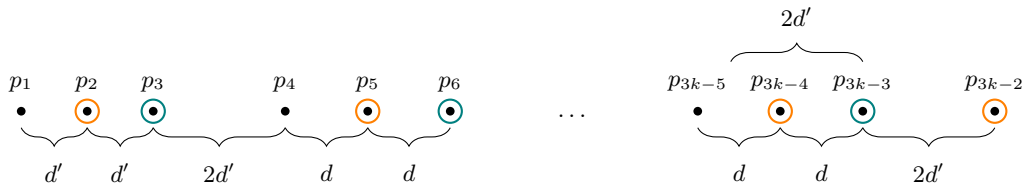


Figure 1 The k-center instance given to an α-approximation algorithm before the adversarial insertion of the point p^+. The orange circles mark the α-approximate placement of the k centers. The teal circles mark the α-approximate placement of $k-1$ centers.

First, we show that there is a unique α-approximate k-center solution. This k-center solution consists of the $k-1$ points p_i with $i \equiv 2 \pmod 3$ and the last point at p_{3k-2} (see Figure 1); the k-center cost of this solution is d. Regarding the uniqueness, each triple p_i, p_{i+1}, p_{i+2} (for every $1 \leq i \leq 3k-5$ with $i \equiv 1 \pmod 3$) and the point p_{3k-2} must contain a center, as otherwise the cost would be at least $2d'$. Moreover, each such center must be placed at the middle point of each triple, as otherwise the cost would be at least $2d'$ (for the leftmost triple) or at least $\min(2d, 2d') = 2d'$ (for any other triple). A strict lower bound on $2d'$ is:

$$2d' > 2\frac{\alpha}{2}d > \alpha d,$$

and thus placing a center at the middle point of each triple and a center at p_{3k-2} is the only possible α-approximate k-center solution.

Assume now that the adversary inserts a point p^+ at a distance greater than $2\alpha d$ from any other point $p_1, p_2, \ldots, p_{3k-2}$. We show that there is a unique α-approximate k-center solution whose cost is $2d'$. To obtain such a solution, one center has to move to p^+, as otherwise the cost would be at least $2\alpha d > \alpha \cdot 2d'$. Hence, we have $k-1$ centers remaining to cover the original $3k-2$ points. To obtain this unique k-center solution, the center from p_{3k-2} is moved to p^+ and all the remaining centers are shifted to the right; namely the $k-1$ remaining centers are placed at p_i where $i \equiv 0 \pmod 3$ (see points marked in teal in Figure 1). The cost of this solution is $2d'$ because $2d < 3d'$ and thus $d < 2d'$; the leftmost point in each triple (except the leftmost one) is covered by the center in the left neighboring triple.

We now show that no other placement of $k-1$ centers among the old points can achieve a cost smaller or equal to $\alpha \cdot 2d'$. Leaving a triple without a center would lead to a cost of at least:

$$\min(4d', 2d' + d) > 2d > \alpha \cdot 2d'.$$

Hence, for any α-approximate solution each triple has to have exactly one center. Suppose to the contrary that in the triple p_i, p_{i+1}, p_{i+2} for some $i \equiv 1 \pmod 3$ the center is not at p_{i+2}. If it is the rightmost triple ($i = 3k-5$), the cost of the solution is at least $2d' + d > 3d' > \alpha \cdot 2d'$.

Otherwise, consider the largest i with $i \equiv 1 \pmod 3$ such that the center for the triple p_i, p_{i+1}, p_{i+2} is not at p_{i+2}. If $i \neq 1$, the cost would be at least $\min(d + 2d', 2d) = 2d > 2\alpha d'$. If $i = 1$, the cost would be at least $\min(3d', 2d) = 2d > 2\alpha d'$.

After the adversarial insertion of p^+, the unique α-approximate solution is given by the set of centers $\{p_i \mid 3 \leq i \leq 3k-3, i \equiv 0 \pmod 3\} \cup \{p^+\}$. As a consequence, the recourse of the solution before and after the adversarial point insertion is k, concluding the proof. ◀

References

1 Kiarash Banihashem, Jeff Giliberti, Samira Goudarzi, MohammadTaghi Hajiaghayi, Peyman Jabbarzade, and Morteza Monemizadeh. Dynamic diameter in high-dimensions against adaptive adversary and beyond, 2025. doi:10.48550/arXiv.2511.01065.

2 MohammadHossein Bateni, Hossein Esfandiari, Hendrik Fichtenberger, Monika Henzinger, Rajesh Jayaram, Vahab Mirrokni, and Andreas Wiese. Optimal fully dynamic k-center clustering for adaptive and oblivious adversaries. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 2677–2727. SIAM, 2023. doi:10.1137/1.9781611977554.CH101.

3 Aaron Bernstein, Sebastian Forster, and Monika Henzinger. A deamortization approach for dynamic spanner and dynamic maximal matching. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1899–1918. SIAM, 2019. doi:10.1137/1.9781611975482.115.

4 Sayan Bhattacharya, Ruoxu Cen, and Debmalya Panigrahi. Fully dynamic set cover: Worst-case recourse and update time. *CoRR*, abs/2511.08485, 2025. To appear in STOC'26. doi:10.48550/arXiv.2511.08485.

5 Sayan Bhattacharya, Martín Costa, and Ermiya Farokhnejad. Fully dynamic k-median with near-optimal update time and recourse. In Michal Koucký and Nikhil Bansal, editors, *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC 2025, Prague, Czechia, June 23-27, 2025*, pages 1166–1177. ACM, 2025. doi:10.1145/3717823.3718176.

6 Sayan Bhattacharya, Martín Costa, Ermiya Farokhnejad, Silvio Lattanzi, and Nikos Parotsidis. Almost optimal fully dynamic k-center clustering with recourse. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL: https://proceedings.mlr.press/v267/bhattacharya25a.html.

7 Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. Incremental clustering and dynamic information retrieval. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 626–635. ACM, 1997. doi:10.1145/258533.258657.

8 Moses Charikar and Shay Solomon. Fully dynamic almost-maximal matching: Breaking the polynomial worst-case time barrier. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, Prague, Czech Republic, July 9-13, 2018*, LIPIcs, pages 33:1–33:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.33.

9 Vincent Cohen-Addad, Chris Schwiegelshohn, and Christian Sohler. Diameter and k-center in sliding windows. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 19:1–19:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.19.

10 Hendrik Fichtenberger, Silvio Lattanzi, Ashkan Norouzi-Fard, and Ola Svensson. Consistent k-clustering for general metrics. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 2660–2678. SIAM, 2021. doi:10.1137/1.9781611976465.158.

11 Sebastian Forster and Antonis Skarlatos. Dynamic consistent k-center clustering with optimal recourse. In Yossi Azar and Debmalya Panigrahi, editors, *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pages 212–254. SIAM, 2025. doi:10.1137/1.9781611978322.7.

12 Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.*, 38:293–306, 1985. doi:10.1016/0304-3975(85)90224-5.

13 Fabrizio Grandoni, Chris Schwiegelshohn, Shay Solomon, and Amitai Uzrad. Maintaining an EDCS in general graphs: Simpler, density-sensitive and with worst-case time bounds. *ACM Trans. Algorithms*, 22(1):10:1–10:16, 2026. doi:10.1145/3765900.

14 Mara Grilnberger and Antonis Skarlatos. Adaptive fully dynamic k-center clustering with (near-)optimal worst-case guarantees, 2026. doi:10.48550/arXiv.2604.01726.

15 Sudipto Guha. Tight results for clustering and summarizing data streams. In Ronald Fagin, editor, *Database Theory - ICDT 2009, 12th International Conference, St. Petersburg, Russia, March 23-25, 2009, Proceedings*, ACM International Conference Proceeding Series, pages 268–275. ACM, 2009. doi:10.1145/1514894.1514926.

16 Dorit S. Hochbaum and David B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. *J. ACM*, 33(3):533–550, 1986. doi:10.1145/5925.5933.

17 Wen-Lian Hsu and George L. Nemhauser. Easy and hard bottleneck location problems. *Discret. Appl. Math.*, 1(3):209–215, 1979. doi:10.1016/0166-218X(79)90044-1.

18 László Kozma. The price of incrementality in k-center clustering, 2026. doi:10.48550/arXiv.2606.08713.

19 Silvio Lattanzi and Sergei Vassilvitskii. Consistent k-clustering. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1975–1984. PMLR, 2017. URL: http://proceedings.mlr.press/v70/lattanzi17a.html.

20 Shay Solomon and Amitai Uzrad. Dynamic set cover with worst-case recourse. *CoRR*, abs/2511.07354, 2025. To appear in ICALP'26. doi:10.48550/arXiv.2511.07354.

Optimal Stable Coresets for Geometric Median via Uniform Sampling

Amir Carmel
Weizmann Institute of Science, Rehovot, Israel

Robert Krauthgamer
The Harry Weinrebe Professorial Chair of Computer Science,
Weizmann Institute of Science, Rehovot, Israel

Nir Petruschka
Weizmann Institute of Science, Rehovot, Israel

Abstract

The geometric median problem asks to find a point in $\mathbb{R}^d$ that minimizes the sum of Euclidean distances to an input set. It is a classical problem in computational geometry and appears as a subroutine in numerous optimization tasks, many of which require the solution to satisfy additional structural constraints. A common approach to reduce the input size is to construct a coreset, which is a small weighted subset that faithfully represents the input for a specific optimization problem. Strong coresets preserve the cost of every candidate solution but require linear time to construct; weak coresets admit sublinear construction, in fact by uniform sampling, but only preserve near-optimal solutions, which is insufficient when the solution is constrained. To address this, we focus instead on the recently introduced intermediate notion of a *stable coreset*, which simultaneously handles all constrained variants. Currently, there is a large gap between the known sample sizes for stable and weak coresets.

Our main result is that a uniform sample of size $O(\epsilon^{-2} \log \frac{1}{\epsilon})$ is a stable $(\epsilon, O(\epsilon))$-coreset for the geometric median, with high constant probability, and this bound is tight up to the logarithmic factor. Our analysis adapts recent machinery of Carmel and Krauthgamer (ICLR 2026) for constructing stable coresets, which incurs an $O(\log d)$ factor. We show an iterative argument that progressively reduces the sample size, and eliminates this dependence on the dimension d. At a high level, this approach resembles the technique of iterative size reduction, which is applicable for strong coresets but not for weak coresets.

2012 ACM Subject Classification Theory of computation → Sketching and sampling; Theory of computation → Streaming models

Keywords and phrases clustering, geometric median, coresets, stable coresets, uniform sampling

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.30

Category APPROX

Related Version *Full Version*: https://arxiv.org/abs/2606.30348

Funding *Robert Krauthgamer*: Work partially supported by the Israel Science Foundation grant #1336/23.

1 Introduction

Clustering is a basic tool for understanding large collections of data, with applications throughout machine learning, statistics, and optimization. The core idea is to partition the data points into sets (called clusters) that share common characteristics, and a classical formulation is center-based clustering, where each cluster is represented by a single point. One of the most fundamental problems in this family is the *geometric median*, which asks for a point that minimizes the sum of distances to all the input points. Despite its


© Amir Carmel, Robert Krauthgamer, and Nir Petruschka;
licensed under Creative Commons License CC-BY 4.0



apparent simplicity, the geometric median exhibits strong robustness properties and has broad applications, from robust statistics to facility location and machine learning. To define this problem formally, we view it as a special case of the *k-median problem*, where the input is a point set P in a metric space $(\mathcal{X}, \mathrm{dist})$, and the goal is to find a set $C \subseteq \mathcal{X}$ of k centers that minimize the objective

$$\mathrm{cost}(C, P) := \sum_{p \in P} \min_{c \in C} \mathrm{dist}(p, c).$$

The *geometric median problem* (also called the *Fermat-Weber problem*) is the case $k = 1$ in the widely studied setting of Euclidean spaces, i.e., $\mathcal{X} = \mathbb{R}^d$ and $\mathrm{dist}(x, y) = \|x - y\|_2$. Our main result addresses the geometric median problem, however the broader context is often useful, e.g., our proof connects this problem to 1-median in ℓ_1 spaces.

As datasets grow in size, data reduction techniques have become essential, as they enable algorithms to operate on small summaries rather than on the entire dataset. A coreset formalizes this idea by providing a small instance of the same problem, where solving the problem on the coreset immediately provides similar guarantees for the original dataset. Coresets usually offer a tunable tradeoff between size and quality, the latter measured as an approximation factor. A *strong ϵ-coreset* for k-median is a weighted subset $Q \subseteq P$ such that for every set of k centers $C \subset \mathbb{R}^d$, the cost is approximately preserved, namely,

$$\mathrm{cost}(C, Q) \in (1 \pm \epsilon) \cdot \mathrm{cost}(C, P). \tag{1}$$

Strong coresets for the classical k-median problem in Euclidean space, alongside numerous variants, have been studied extensively over the past two decades, see Section 1.3 for more details.

Many modern applications involve massive datasets, for which even linear-time processing might be prohibitive. Yet, constructing a strong coreset inherently requires reading each data point, which takes at least linear time $\Omega(nd)$, regardless of the coreset's size.[1] Therefore, to open the door for sublinear-time algorithms, one natural relaxation is the notion of a *weak coreset*. A weighted subset $Q \subseteq P$ is called a weak (ϵ, η)-coreset if every near-optimal solution for Q is also a near-optimal solution for P, namely, for every set of k centers $C \subset \mathbb{R}^d$,

$$\text{if } \mathrm{cost}(C, Q) \le (1+\epsilon) \cdot \min_{|C'|=k} \mathrm{cost}(C', Q), \text{ then } \mathrm{cost}(C, P) \le (1+\eta) \cdot \min_{|C'|=k} \mathrm{cost}(C', P). \tag{2}$$

It is worth noting that most constructions achieve $\eta = O(\epsilon)$, but covering the entire literature requires two separate parameters. Weak coresets can be constructed via uniform sampling, a method widely adopted in practice, often without theoretical guarantees, and naturally suited to streaming and distributed settings. See Table 1 for known bounds and references.

Despite its appeal, the weak coreset notion has a fundamental limitation: A strong coreset has guarantees for *every* candidate solution, but a weak coreset provides guarantees only for candidates that are near-optimal (on the coreset). This might be a severe limitation when the solutions of interest are constrained to a subset $\mathcal{X}' \subset \mathcal{X}$ (because an optimal solution in $\mathcal{X}'$ need not be optimal also in $\mathcal{X}$), and such constraints indeed arise from structural requirements or downstream tasks, as we discuss further below.

[1] Consider an instance P with a single extremely distant point. A strong coreset must include this point, which requires scanning the entire instance.

Table 1 Sublinear-time constructions of coresets for the geometric median problem. In all of them, the sample complexity equals the (listed) coreset size; in fact, the coreset is simply the sample, except when marked by †, in which the sample is post-processed. Strong coresets are not listed as they require at least linear time.

Coreset type	Coreset size	Reference
weak $(0, \epsilon)$	$\widetilde{O}(\epsilon^{-2}d)$	[1, 45]
	$\widetilde{O}(\epsilon^{-4})$	[25]
weak $(\epsilon, O(\epsilon))$	$\widetilde{O}(\epsilon^{-4})$ †	[21]
	$\widetilde{O}(\epsilon^{-3})$	[33]
	$\widetilde{O}(\epsilon^{-2})$	[53]
stable $(\epsilon, O(\epsilon))$	$\widetilde{O}(\epsilon^{-2}\log d)$	[13]
	$\widetilde{O}(\epsilon^{-2})$	Theorem 1.1

The notion of a *stable coreset*, recently introduced by [13], is intermediate between the strong and weak coreset notions and addresses this issue, by preserving the relative quality of *all pairs* of candidate solutions. Formally, a weighted subset $Q \subseteq P$ is called a stable (ϵ, η)-coreset if for every pair of candidates $C, C' \subset \mathbb{R}^d$,

$$\text{if } \operatorname{cost}(C, Q) \le (1+\epsilon)\cdot \operatorname{cost}(C', Q), \text{ then } \operatorname{cost}(C, P) \le (1+\eta)\cdot \operatorname{cost}(C', P). \tag{3}$$

Crucially, since this guarantee holds simultaneously for all pairs of candidates, the same stable coreset can be reused across multiple constraints and downstream tasks, and its construction does not require a priori knowledge of the constraints.

The geometric median problem is used as a subroutine in numerous optimization tasks, and many of them require the solution to satisfy additional constraints, which essentially restrict it to a predetermined subset $\mathcal{X}' \subset \mathcal{X} = \mathbb{R}^d$. For instance, the solution may be required to lie in a low-dimensional linear subspace, or more generally on an algebraic variety, or to have a sparse coordinate structure (e.g., $\|c\|_0 \le \alpha$).

We illustrate this with three concrete examples. First, in kernel-based methods, the input points are all images of a feature map φ and also the solution must be an image of φ [44, 47]; in this case, $\mathcal{X}'$ is simply the image set of φ. Second, the *discrete* geometric median problem requires the solution to be one of the input points in P [46, 30, 24]; in this case $\mathcal{X}' = P$ is discrete. Third, the *ultrametric* 1-*median* problem, in which the input distances form an ultrametric and the solution is a median terminal node, admits a $(1+\epsilon)$-approximation in $\widetilde{O}(\epsilon^{-3})$ time [14]. Since finite ultrametrics embed isometrically into ℓ_2 [51], this is a special case of constrained ℓ_2 1-median.

In all these cases where we seek an optimal or near-optimal solution in $\mathcal{X}'$, a weak coreset provides no guarantees, while a stable coreset is applicable, in fact simultaneously for all possible restrictions $\mathcal{X}'$.

1.1 Our Results

A long line of work has progressively refined the size of weak coresets (see Table 1), however these results and techniques do not extend (at least not directly) to stable coresets, for which the known bound falls behind and even depends on the dimension d. Our main result is a stable coreset for geometric median that is constructed via uniform sampling (with no post-processing) and has near-optimal size and query complexity. In our result, a uniform sample is one drawn without replacement.

▶ **Theorem 1.1.** *Let $\epsilon \in (0, 1/4)$ and let $P \subset \mathbb{R}^d$ be finite. Then a uniform sample of size $O(\epsilon^{-2} \log \frac{1}{\epsilon})$ from P is a stable $(\epsilon, 17\epsilon)$-coreset for 1-median in ℓ_2^d with probability at least $4/5$.*

This result improves over the stable coreset of [13], which is also via uniform sampling and has size $\widetilde{O}(\epsilon^{-2} \log d)$. Our bound completely eliminates the dependence on the dimension. Technically, their main result is for ℓ_1 metrics, and the Euclidean case follows by Dvoretzky's theorem [29, 49]. Previously, a similar coreset size $\widetilde{O}(\epsilon^{-2})$, also via uniform sampling, was known only for weak coreset [53], which is strictly less powerful. Technically, it is obtained as a corollary of a result for multiple ℓ_p regression, whose proof leverages ℓ_p Lewis weight sampling. In contrast, our proof refines the framework for stable coresets recently developed by [13], and employs an iterative argument to progressively reduce the sample size, see Section 1.2.

The dependence on ϵ in our bound is near-tight, as an $\Omega(\epsilon^{-2})$ lower bound follows from more general results of [16, 48]. For completeness, we provide a short self-contained proof in Appendix A .

▶ Remark 1.2. We consider only the case $k = 1$ in order to obtain clean theoretical guarantees. It is well-known that sublinear algorithms face a major challenge when $k > 1$, because small clusters can be missed by sublinear methods. Prior work by [33] (see also references therein) has formalized this issue through a *balancedness parameter* $\beta \in (0, 1]$, which measures the smallest cluster size in an optimal solution, modifying the coreset definition to consider only β-balanced clusterings of the coreset.

1.2 Technical Overview

We now overview the proof of our main result, Theorem 1.1. Our starting point is a result of [13, Corollary D.2], which establishes that a uniform sample of size $O(\epsilon^{-2} \log(d/\epsilon))$ is a stable $(\epsilon, O(\epsilon))$-coreset for 1-median in ℓ_2^d with high constant probability. Their proof exploits the fact that ℓ_2^d embeds into $\ell_1^{O(d)}$, by Dvoretzky's theorem, and applies their analysis for the 1-median problem in ℓ_1. We make a simple observation, that a folklore terminal embedding (Theorem 2.3) reduces the problem to the case where $d \le n$, and therefore a uniform sample Q of size $O(\epsilon^{-2} \log(n/\epsilon))$ suffices for such a coreset, regardless of the ambient dimension d of our dataset P.

Building on this simple observation, our high-level idea is to apply iterative size reduction, a technique introduced by [9] for strong coresets, hence we need to adapt it to stable coresets. Suppose we have a uniform sample $Q \subset P$ of size $\widetilde{O}(\epsilon^{-2} \log n)$, and then we pick from it a uniform sample $R \subseteq Q$ of size $\widetilde{O}(\epsilon^{-2} \log |Q|) = \widetilde{O}(\epsilon^{-2} \log \log n)$. By our preceding discussion, with constant high probability, Q is a stable coreset of P, and also R is a stable coreset of Q. Now if coresets can be composed (i.e., a coreset of a coreset is a coreset), which is indeed the case for strong coresets, then R is also a coreset of our initial dataset P. The crux is to apply this argument iteratively until the size of the sample no longer decreases, and by solving $|Q| \le \epsilon^{-2} \log(|Q|/\epsilon)$ we can easily see that the final sample size is $\widetilde{O}(\epsilon^{-2})$. Importantly, while this analysis goes through cumbersome iterations, the algorithm is simple: A uniform sample R from Q, which itself is a uniform sample from P, amounts to a uniform sample R taken directly from P. Therefore, the final coreset is just a uniform sample of size $\widetilde{O}(\epsilon^{-2})$ from the input P.

Unfortunately, the above strategy has considerable gaps. One issue is that each iteration might fail with a constant probability, which is too large because the number of iterations is typically $O(\log^* n)$. One can decrease this probability by tuning parameters differently

(standard amplification by majority over repetitions would deviate from uniform sampling), but the final coreset size will depend on n, missing our intended goal. Another issue is that the errors accumulate over the iterations. Prior work [9, 21, 37] addressed this by setting a different approximation parameter ϵ_i for each iteration i. Typically, one starts with a small value, say $\epsilon_1 = 1/\log n$, and increases it exponentially fast until some final value ϵ_{final}, which determines the final coreset size. For strong coresets, the accumulated error is $\prod_i(1+\epsilon_i) \le 1 + \sum_i O(\epsilon_i) \le 1 + O(\epsilon_{\text{final}})$, however for stable coresets the errors compound completely differently. Suppose that each iteration yields a stable $(\epsilon, 2\epsilon)$-coreset, then we must set $\epsilon_{i+1} = \epsilon_i/2 = \epsilon_1/2^i$, which results overall in a stable $(\epsilon_{\text{final}}, \epsilon_1)$-coreset. As the number of iterations grows with n, albeit slowly like $O(\log^* n)$, this guarantee is rather poor and certainly misses our intended goal.

Since iterative size reduction cannot be applied in a black-box manner, we need to implement the iterations in a more specialized manner. We identify that (a slight adaptation of) a notion called *Relative Cost-Difference Approximation* (RCDA), that was introduced by [13], avoids the core issue in the scheme above. Informally, the RCDA notion asserts that the difference in costs between any two candidate centers is approximately the same when measured on the sample versus on the full dataset. We compose this RCDA notion over the iterations, which requires some technical adaptations. It also introduces some error terms and failure probabilities, and tracking these carefully over the iterations is somewhat intricate. Our main technical lemma (Lemma 4.2) shows that the final sample is an RCDA for P, which, in turn, implies that it forms a stable coreset.

1.3 Related Work

Strong coresets, which are often simply referred to as coresets, have been extensively studied for the k-median and k-means problems in Euclidean metrics. After a long line of works, which includes [31, 15, 28, 52, 39, 9, 22], the current state-of-the-art constructions for k-median and k-means match the known lower bound of $\Omega(k\epsilon^{-2})$, up to polylogarithmic factors [20, 37]. For the ℓ_1 metric, a coreset of size $\text{poly}(k/\epsilon)$ follows from [40] by going through ℓ_2-squared. The case of $k = 1$ has received special attention. In low-dimensional Euclidean spaces [32] obtain a strong coreset of size $\widetilde{O}(\sqrt{d}/\epsilon)$, which was subsequently improved by [2] to $\widetilde{O}(\epsilon^{-d/(d+1)})$ for any constant dimension d.

Many coreset variants of the k-median and k-means problems have been studied as well. Closest to our work is the line of weak coresets for 1-median in Euclidean spaces, where the construction can be performed in sublinear time; see Table 1 for a detailed comparison of previous results. Additionally, coresets have been studied for other variants of the geometric median problem and the more general k-median problem in Euclidean metrics, including capacitated k-median [7, 36], fair clustering [50, 35, 18], robust clustering [34, 41, 27], fuzzy clustering [6], ordered weighted clustering [8], and time-series clustering [38].

Another closely related problem is the 1-center, also known as the minimum enclosing ball problem. Strong coresets for this problem in Euclidean metrics require exponential dependence on the dimension [3]. Weak coresets, in contrast, were studied by [11, 10], who show a tight size bound of $\lceil 1/\epsilon \rceil$. For the ℓ_1 metric, [12] studied both strong and weak coresets, and showed that the coreset size must depend on the dimension.

Lastly, a separate line of work studies sublinear approximation algorithms for the geometric median (or the related 1-means objective), rather than constructing a coreset [19, 5].

2 Preliminaries

In the 1-median problem on the metric space $\ell_q^d = (\mathbb{R}^d, \|\cdot\|_q)$, we are given a set $P \subset \mathbb{R}^d$ and the goal is to find a point $x \in \mathbb{R}^d$ that minimizes the sum of distances to P, that is, to minimize the cost function

$$\mathrm{cost}_q(x, P) = \frac{1}{|P|} \sum_{p \in P} \|x - p\|_q .$$

We denote by μ^P an optimal solution to the median problem on P, $\mu^P = \mathrm{argmin}_{x \in \mathbb{R}^d} \mathrm{cost}_q(x, P)$.

In this paper, we focus on the 1-median problem in the Euclidean metric, that is, $q = 2$. Throughout our analysis, we leverage metric embeddings to translate results between ℓ_2 and ℓ_1.

The following theorem is a classical consequence of Gordon's refinement of Dvoretzky's theorem [29, 49].

▶ **Theorem 2.1** (Dvoretzky's theorem). *For every $\gamma \in (0, 1)$ and natural number d, there exists a linear map $f : \mathbb{R}^d \to \mathbb{R}^m$ with $m = O(\gamma^{-2} d)$ such that*

$$\forall x \in \mathbb{R}^d, \qquad \|f(x)\|_1 \in (1 \pm \gamma) \|x\|_2 .$$

Given such embedding f, it immediately follows that the cost function is also preserved up to $1 \pm \gamma$. For a set $P \subset \mathbb{R}^d$, we denote by $f(P) = \{f(p) : p \in P\}$ the image of P under f.

▶ **Corollary 2.2.** *Let $f : \mathbb{R}^d \to \mathbb{R}^m$ be as in Theorem 2.1 with parameter γ and let $P \subset \mathbb{R}^d$. Then*

$$\forall x \in \mathbb{R}^d, \qquad \mathrm{cost}_1(f(x), f(P)) \in (1 \pm \gamma) \mathrm{cost}_2(x, P).$$

We will need the following folklore isometric terminal embedding in ℓ_2. Although variants of it appear in [26, 39], we include a proof for completeness.

▶ **Theorem 2.3** (Folklore). *Let $P \subset \mathbb{R}^d$ of size n. Then there exists an isometric terminal embedding $f : \mathbb{R}^d \to \mathbb{R}^n$, that is, for all $x \in \mathbb{R}^d$ and $p \in P$, $\|f(x) - f(p)\|_2 = \|x - p\|_2$.*

Proof. Let $P = \{p_1, \ldots, p_n\} \subset \mathbb{R}^d$ be a terminal set. By shifting the points, we can assume w.l.o.g. that p_n lies at the origin. Consider the $(n-1)$-dimensional subspace V spanned by the vectors $p_1, \ldots, p_{n-1}$. As every d-dimensional subspace of ℓ_2 is isometric to $\mathbb{R}^d$, we can also assume w.l.o.g. that $V = \mathbb{R}^{n-1}$. Let $\Pi : \mathbb{R}^d \to V$ be the orthogonal projection map onto V and define the terminal embedding $f : \mathbb{R}^d \to V \times \mathbb{R} = \mathbb{R}^n$ by $f(z) = (\Pi(z), \|\Pi(z) - z\|_2)$. Using the Pythagorean theorem, we conclude that for every $p \in P \subset V$ and $x \in \mathbb{R}^d$

$$\begin{aligned} \|f(p) - f(x)\|_2^2 &= \|\Pi(p) - \Pi(x)\|_2^2 + (\|\Pi(x) - x\|_2 - 0)^2 \\ &= \|p - \Pi(x)\|_2^2 + \|\Pi(x) - x\|_2^2 = \|p - x\|_2^2. \end{aligned}$$

◀

3 RCDA in ℓ_2 via Dimension Reduction

The RCDA property was formulated in [13]. Here we introduce a slightly modified definition.

▶ **Definition 3.1** (Relative Cost-Difference Approximation). *We say that $Q \subseteq P$ is an ϵ-RCDA$_q$ of P if*

$$\forall x, y \in \mathbb{R}^d, \quad \Big| \big[\mathrm{cost}_q(x, P) - \mathrm{cost}_q(y, P)\big] - \big[\mathrm{cost}_q(x, Q) - \mathrm{cost}_q(y, Q)\big] \Big| \leq \epsilon \cdot (\mathrm{cost}_q(x, P) + \mathrm{cost}_q(y, P)). \tag{4}$$

We will denote the left-hand side by $\mathrm{RCDA}_q(x, y, Q, P)$ *for brevity.*

Our definition differs from that in [13] in that (4) quantifies over all pairs $x, y \in \mathbb{R}^d$, whereas the original property was defined with respect to x and the median μ^P. Specifically, the original definition required

$$\forall x \in \mathbb{R}^d, \qquad \mathrm{RCDA}_q(x, \mu^P, Q, P) \le \epsilon \, \mathrm{cost}_q(x, P). \tag{5}$$

However, the two definitions are essentially equivalent. Indeed, if for every $x \in \mathbb{R}^d$, $\mathrm{RCDA}_q(x, \mu^P, Q, P) \le \epsilon \cdot \mathrm{cost}_q(x, P)$, then for every $x, y \in \mathbb{R}^d$,

$$\mathrm{RCDA}_q(x, y, Q, P) \le \mathrm{RCDA}_q(x, \mu^P, Q, P) + \mathrm{RCDA}_q(\mu^P, y, Q, P) \le \epsilon \cdot (\mathrm{cost}_q(x, P) + \mathrm{cost}_q(y, P)),$$

where the first inequality follows from the triangle inequality. Conversely, if for every $x, y \in \mathbb{R}^d$, we have $\mathrm{RCDA}_q(x, y, Q, P) \le \epsilon \cdot (\mathrm{cost}_q(x, P) + \mathrm{cost}_q(y, P))$, then for every $x \in X$,

$$\mathrm{RCDA}_q(x, \mu^P, Q, P) \le \epsilon \cdot (\mathrm{cost}_q(x, P) + \mathrm{cost}_q(\mu^P, P)) \le 2\epsilon \cdot \mathrm{cost}_q(x, P).$$

The main technical result from [13] establishes the RCDA property for a uniform sample in ℓ_1^d.

▶ **Lemma 3.2** (Lemma 4.6 in [13]). *Let $\epsilon \in (0, 1)$ and let $P \subset \mathbb{R}^d$ of size n. Then a uniform sample $Q \subseteq P$ of size $O(\epsilon^{-2} \log \frac{d}{\delta})$ is an ϵ-RCDA$_1$ of P in ℓ_1^d with probability at least $1 - \delta$.*

▶ Remark 3.3. While [13] does not explicitly state whether the uniform sample is taken with or without replacement, their result holds in both cases. The only step in their proof that depends on the sampling scheme is a theorem from [42] on ϵ-approximations, which applies equally to sampling with and without replacement [23].

A direct consequence of the RCDA property, is a uniform bound over all costs.

▶ **Proposition 3.4.** *Let $\epsilon \in (0, 1/2)$ and $\alpha \ge 1$. If Q is an ϵ-RCDA$_q$ of P and $\mathrm{cost}_q(\mu^P, Q) \le \alpha \, \mathrm{cost}_q(\mu^P, P)$, then,*

$$\forall x \in \mathbb{R}^d, \qquad \mathrm{cost}_q(x, Q) \le 2\alpha \, \mathrm{cost}_q(x, P).$$

Proof. By Definition 3.1, we have $\mathrm{RCDA}_q(x, \mu^P, Q, P) \le \epsilon \cdot (\mathrm{cost}_q(x, P) + \mathrm{cost}_q(\mu^P, P))$. Therefore:

$$\begin{aligned}
\mathrm{cost}_q(x, Q) &\le \mathrm{cost}_q(x, P) + \mathrm{cost}_q(\mu^P, Q) - \mathrm{cost}_q(\mu^P, P) + \epsilon(\mathrm{cost}_q(x, P) + \mathrm{cost}_q(\mu^P, P)) \\
&\le (1 + \epsilon) \, \mathrm{cost}_q(x, P) + (\alpha - 1 + \epsilon) \, \mathrm{cost}_q(\mu^P, P) \\
&\le (1 + \epsilon) \, \mathrm{cost}_q(x, P) + (\alpha - 1 + \epsilon) \, \mathrm{cost}_q(x, P) && (\mu^P \text{ is optimal}) \\
&= (\alpha + 2\epsilon) \, \mathrm{cost}_q(x, P) \le 2\alpha \, \mathrm{cost}_q(x, P). && (2\epsilon \le 1 \le \alpha)
\end{aligned}$$

◀

We now establish that uniform sampling yields RCDA in ℓ_2^d by reducing to the ℓ_1 case.

▶ **Lemma 3.5.** *Let $\epsilon \in (0, 1)$, $\alpha \ge 1$, and let $P \subset \mathbb{R}^d$ of size n. Then, with probability at least $1 - 1/\alpha - \delta$, a uniform sample $Q \subseteq P$ of size $O(\epsilon^{-2} \log \frac{\alpha \cdot n}{\epsilon \cdot \delta})$ satisfies:*

1. *Q is an ϵ-RCDA$_2$ of P in ℓ_2^d, and,*
2. *For every $x \in \mathbb{R}^d$, $\mathrm{cost}_2(x, Q) \le 3\alpha \, \mathrm{cost}_2(x, P)$.*

Proof. By Theorem 2.3, we may assume without loss of generality that $d = n$.

Let $\gamma = \frac{\epsilon}{10\alpha}$. By Theorem 2.1, there exists $m = O(\alpha^2 \epsilon^{-2} n)$ and a linear map $f : \mathbb{R}^d \to \mathbb{R}^m$ such that for all $x \in \mathbb{R}^d$, $\|f(x)\|_1 \in (1 \pm \gamma) \|x\|_2$.

By Lemma 3.2, a uniform sample $Q \subseteq P$ of size $O(\epsilon^{-2} \log \frac{m}{\delta}) = O(\epsilon^{-2} \log \frac{\alpha n}{\epsilon\delta})$ is a $\frac{\epsilon}{2}$-RCDA$_1$ of $f(P)$ in ℓ_1^m with probability at least $1-\delta$. That is,

$$\forall x, y \in \mathbb{R}^d, \qquad \mathrm{RCDA}_1(f(x), f(y), f(Q), f(P)) \\ \leq \frac{\epsilon}{2}(\mathrm{cost}_1(f(x), f(P)) + \mathrm{cost}_1(f(y), f(P))). \tag{6}$$

Let μ^P be the optimal median of P in ℓ_2^d. Since $\mathbb{E}_Q[\mathrm{cost}_1(f(\mu^P), f(Q))] = \mathrm{cost}_1(f(\mu^P), f(P))$, by Markov's inequality, with probability at least $1 - 1/\alpha$,

$$\mathrm{cost}_1(f(\mu^P), f(Q)) \leq \alpha \, \mathrm{cost}_1(f(\mu^P), f(P)). \tag{7}$$

In this event, by Proposition 3.4 applied in ℓ_1^m (noting $\epsilon/2 < 1/2$),

$$\forall x \in \mathbb{R}^d, \qquad \mathrm{cost}_1(f(x), f(Q)) \leq 2\alpha \, \mathrm{cost}_1(f(x), f(P)). \tag{8}$$

Using Corollary 2.2, for all $x \in \mathbb{R}^d$,

$$\begin{aligned}\mathrm{cost}_2(x, Q) &\leq (1+\gamma)\,\mathrm{cost}_1(f(x), f(Q)) \leq (1+\gamma) \cdot 2\alpha \, \mathrm{cost}_1(f(x), f(P)) \\ &\leq \frac{2\alpha(1+\gamma)}{1-\gamma}\,\mathrm{cost}_2(x, P) \leq 3\alpha \, \mathrm{cost}_2(x, P),\end{aligned}$$

where the last inequality uses $\gamma = \epsilon/(10\alpha) \leq 1/10$, establishing property (2).

To show property (1), let $x, y \in \mathbb{R}^d$. Again using Corollary 2.2 applied to each of the four cost terms and the triangle inequality:

$$\begin{aligned}\mathrm{RCDA}_2(x, y, Q, P) \leq \mathrm{RCDA}_1(f(x), f(y), f(Q), f(P)) + \gamma(&\mathrm{cost}_1(f(x), f(P)) + \mathrm{cost}_1(f(y), f(P)) \\ &+ \mathrm{cost}_1(f(x), f(Q)) + \mathrm{cost}_1(f(y), f(Q)))\end{aligned}$$

By the union bound (6) and (8) holds with probability at least $1 - 1/\alpha - \delta$. We get:

$$\begin{aligned}\mathrm{RCDA}_2(x, y, Q, P) &\leq \frac{\epsilon}{2}(\mathrm{cost}_1(f(x), f(P)) + \mathrm{cost}_1(f(y), f(P))) \\ &\quad + \gamma(2\alpha+1)(\mathrm{cost}_1(f(x), f(P)) + \mathrm{cost}_1(f(y), f(P))) \\ &\leq \frac{1}{1-\gamma}\left(\frac{\epsilon}{2} + \gamma(2\alpha+1)\right)(\mathrm{cost}_2(x, P) + \mathrm{cost}_2(y, P)) \\ &\leq \frac{10}{9}\left(\frac{\epsilon}{2} + \frac{\epsilon(2\alpha+1)}{10\alpha}\right)(\mathrm{cost}_2(x, P) + \mathrm{cost}_2(y, P)) \\ &\leq \frac{10}{9}\left(\frac{\epsilon}{2} + \frac{3\epsilon}{10}\right)(\mathrm{cost}_2(x, P) + \mathrm{cost}_2(y, P)) \leq \epsilon(\mathrm{cost}_2(x, P) + \mathrm{cost}_2(y, P)). \blacktriangleleft\end{aligned}$$

4 Upper Bound (Proof of Theorem 1.1)

To apply the iterative size reduction technique, we first establish the following composition property for RCDA, which allows errors to be tracked across iterations.

▶ **Proposition 4.1.** *Let $Q'' \subseteq Q' \subseteq Q$ with $\epsilon, \epsilon' \in (0,1)$ and $\beta \geq 1$. Suppose Q' is an ϵ-RCDA$_q$ of Q and Q'' is an ϵ'-RCDA$_q$ of Q'. If for every $x \in \mathbb{R}^d$ we have $\mathrm{cost}_q(x, Q') \leq \beta \, \mathrm{cost}_q(x, Q)$, then, Q'' is an $(\epsilon + \epsilon'\beta)$-RCDA$_q$ of Q.*

Proof. For $x, y \in \mathbb{R}^d$, using the triangle inequality,

$$\begin{aligned}\mathrm{RCDA}_q(x, y, Q, Q'') &\leq \mathrm{RCDA}_q(x, y, Q, Q') + \mathrm{RCDA}_q(x, y, Q', Q'') \\ &\leq \epsilon(\mathrm{cost}_q(x, Q) + \mathrm{cost}_q(y, Q)) + \epsilon'(\mathrm{cost}_q(x, Q') + \mathrm{cost}_q(y, Q')) \\ &\leq (\epsilon + \epsilon'\beta)(\mathrm{cost}_q(x, Q) + \mathrm{cost}_q(y, Q)). \blacktriangleleft\end{aligned}$$

We are now ready to prove our main technical lemma. The proof involves balancing the errors with the guaranteed size reduction. To streamline the presentation, we use explicit constants and while we attempt to avoid huge constants, they are probably far from optimal.

▶ **Lemma 4.2.** *Let $\epsilon \in (0, \frac{1}{4})$ and let $P \subset \mathbb{R}^d$ of size n. Then, with probability at least $1 - \frac{1}{30}$, a uniform sample $Q \subseteq P$ of size $O(\epsilon^{-2} \log \frac{1}{\epsilon})$ is an ϵ-RCDA$_2$ of P in ℓ_2^d.*

Proof. By Theorem 2.3, we may assume $d = n$. Denote the i-th iterated logarithm (with base 2) by $\log^{(i)}$, i.e., $\log^{(i+1)} n = \log(\log^{(i)} n)$ where $\log^{(0)} n = n$. Let $t \geq 1$ be the smallest integer such that $\log^{(t-1)} n \leq 2048\epsilon^{-2} \log \epsilon^{-1}$; then it also satisfies $\log^{(t-1)} n > \log(2048\epsilon^{-2} \log \epsilon^{-1}) \geq 16$. For $i = 1, \ldots, t$, define

$$\epsilon_i = \frac{\epsilon}{1440(\log^{(i)} n)^{\frac{1}{2}}}, \qquad \alpha_i = 180(\log^{(i)} n)^{\frac{1}{4}}, \qquad \delta_i = 1/\alpha_i.$$

We construct $Q_t \subseteq Q_{t-1} \subseteq \cdots \subseteq Q_1 \subseteq Q_0 = P$, where each Q_{i+1} is obtained from Q_i by uniform sampling of size

$$O\left(\epsilon_{i+1}^{-2} \log\left(\frac{\alpha_{i+1} n_i}{\epsilon_{i+1}\delta_{i+1}}\right)\right)$$

with $n_i = |Q_i|$, as guaranteed by Lemma 3.5. We next prove the following claim.

▷ Claim 4.3. For every $i \in \{1, \ldots, t\}$, with probability at least $1 - 3\sum_{j=1}^{i} \delta_j$,

$$Q_i \text{ is an } \epsilon \sum_{j=1}^{i} \frac{1}{4(\log^{(j)} n)^{\frac{1}{4}}}\text{-RCDA}_2 \text{ of } P.$$

Proof. The proof proceeds by induction on $i \in \{1, \ldots, t\}$.

Base case. By Lemma 3.5 with parameters $\epsilon_1, \alpha_1, \delta_1$, it follows that Q_1 is an ϵ_1-RCDA$_2$ of P with probability at least $1 - 2\delta_1$. The base case holds as $\epsilon_1 = \epsilon/1440(\log^{(1)} n)^{\frac{1}{2}} \leq \epsilon/4(\log^{(1)} n)^{\frac{1}{4}}$.

Inductive step. Since Q_i is a uniform sample of P, by Markov's inequality, with probability at least $1 - 1/\alpha_{i+1}$,

$$\text{cost}_2(\mu^P, Q_i) \leq \alpha_{i+1} \text{cost}_2(\mu^P, P).$$

By Proposition 3.4, for every $x \in \mathbb{R}^d$, $\text{cost}_2(x, Q_i) \leq 2\alpha_{i+1} \text{cost}_2(x, P)$. Thus, applying Lemma 3.5 to sampling Q_{i+1} from Q_i gives that, with probability at least $1-2\delta_{i+1}$, Q_{i+1} is an ϵ_{i+1}-RCDA$_2$ of Q_i. As by the induction hypothesis, Q_i is an $\epsilon \sum_{j=1}^{i} \frac{1}{4(\log^{(j)} n)^{\frac{1}{4}}}$-RCDA$_2$ of P with probability at least $1-3\sum_{j=1}^{i} \delta_j$, both events hold with probability at least $1-3\sum_{j=1}^{i+1} \delta_j$ by a union bound. Assuming they both hold, we can now apply Proposition 4.1 and get,

$$Q_{i+1} \text{ is an } \left(\epsilon \sum_{j=1}^{i} \frac{1}{4(\log^{(j)} n)^{\frac{1}{4}}} + \epsilon_{i+1} \cdot 2\alpha_{i+1}\right)\text{-RCDA}_2 \text{ of } P.$$

Since $\epsilon_{i+1} \cdot 2\alpha_{i+1} = \frac{\epsilon}{1440(\log^{(i+1)} n)^{\frac{1}{2}}} \cdot 360(\log^{(i+1)})^{\frac{1}{4}} = \frac{\epsilon}{4(\log^{(i+1)} n)^{\frac{1}{4}}}$, we have,

$$\epsilon \sum_{j=1}^{i} \frac{1}{4(\log^{(j)} n)^{\frac{1}{4}}} + \epsilon_{i+1} \cdot 2\alpha_{i+1} \leq \epsilon \sum_{j=1}^{i+1} \frac{1}{4(\log^{(j)} n)^{\frac{1}{4}}},$$

concluding the induction step, and proving the claim. ◁

We proceed with the proof of Lemma 4.2. By choice of t, for $1 \le j \le t-2$, we have $(\log^{(j+1)} n)^{\frac{1}{4}} \le \frac{1}{8}(\log^{(j)} n)^{\frac{1}{4}}$ since $\log^{(j)} n \ge 2^{16}$. Thus,

$$\epsilon \sum_{j=1}^{t-1} \frac{1}{4(\log^{(j)} n)^{\frac{1}{4}}} \le \frac{\epsilon}{4(\log^{(t-1)} n)^{\frac{1}{4}}} \sum_{j=0}^{t-2} 8^{-j} \le \frac{2\epsilon}{7(\log^{(t-1)} n)^{\frac{1}{4}}},$$

where the last inequality uses the standard bound over a sum of a geometric series. Additionally, since $\log^{(t-1)} n \ge 16$,

$$\epsilon \sum_{j=1}^{t} \frac{1}{4(\log^{(j)} n)^{\frac{1}{4}}} \le \frac{2\epsilon}{7(\log^{(t-1)} n)^{\frac{1}{4}}} + \frac{\epsilon}{4(\log^{(t)} n)^{\frac{1}{4}}} \le \frac{\epsilon}{2},$$

and similarly,

$$3\sum_{j=1}^{t} \delta_j \le 3\sum_{j=1}^{t-1} \delta_j + 3\delta_t \le \frac{24}{7}\delta_{t-1} + 3\delta_t \le \frac{24}{7 \cdot 180(\log^{(t-1)} n)^{\frac{1}{4}}} + \frac{3}{180(\log^{(t)} n)^{\frac{1}{4}}} \le \frac{1}{105} + \frac{1}{60} \le \frac{3}{100},$$

where overall, we get that Q_t is ϵ-RCDA$_2$ of P with probability at least $1 - \frac{3}{100}$.

Sample size. It is easy to verify the invariant $n_i \le C\,\epsilon^{-2} \log \epsilon^{-1} (\log^{(i)} n)^2$ for all $i \in \{1, \ldots, t\}$, for a sufficiently large constant C. Thus, at iteration t, since $\log^{(t-1)} n = O(\epsilon^{-2} \log \epsilon^{-1})$, we have that $n_t = O(\epsilon^{-2} \log \epsilon^{-1} \cdot (\log(\epsilon^{-2} \log \epsilon^{-1}))^2) = O(\epsilon^{-2} \log^3 \frac{1}{\epsilon})$.

To further improve the sample size to $O(\epsilon^{-2} \log \frac{1}{\epsilon})$, we apply one additional iteration with constant parameters. Set $\epsilon_f = \frac{\epsilon}{4000}$, $\alpha_f = 1000$, and $\delta_f = 1/\alpha_f$, and uniformly sample $Q \subseteq Q_t$ of size

$$O\left(\epsilon_f^{-2} \log \frac{\alpha_f n_t}{\epsilon_f \delta_f}\right) = O(\epsilon^{-2} \log \frac{1}{\epsilon}),$$

where note that, since sampling is uniform at every step, Q is also a uniform sample of P.

Finally, by applying the same arguments as in the inductive step, we get that Q is an

$$\left(\frac{\epsilon}{2} + \epsilon_f \cdot 2\alpha_f\right)\text{-RCDA}_2 = \epsilon\text{-RCDA}_2$$

uniform sample of P of size $O(\epsilon^{-2} \log \frac{1}{\epsilon})$ with probability at least $1 - \frac{3}{100} - (2/\alpha_f + \delta_f) \ge 1 - \frac{1}{30}$. This concludes the proof of Lemma 4.2. ◀

We can now prove our main result, by plugging our notion of RCDA into the framework of [13].

▶ **Theorem 1.1.** *Let $\epsilon \in (0, 1/4)$ and let $P \subset \mathbb{R}^d$ be finite. Then a uniform sample of size $O(\epsilon^{-2} \log \frac{1}{\epsilon})$ from P is a stable $(\epsilon, 17\epsilon)$-coreset for 1-median in ℓ_2^d with probability at least $4/5$.*

Proof. By Lemma 4.2, a uniform sample Q of size $O(\epsilon^{-2} \log \epsilon^{-1})$ is $\frac{\epsilon}{2}$-RCDA$_2$ of P in ℓ_2^d with probability at least $29/30$. By Markov's inequality, with probability at least $5/6$, $\mathrm{cost}_2(\mu^P, Q) \le 6\,\mathrm{cost}_2(\mu^P, P)$. By the union bound, both events occur with probability at least $4/5$. Now, let $x, y \in \mathbb{R}^d$ such that $\mathrm{cost}_2(x, Q) \le (1+\epsilon)\,\mathrm{cost}_2(y, Q)$, then

$$\begin{aligned}
\operatorname{cost}_2(x,P) &\le \tfrac{\epsilon}{2}[\operatorname{cost}_2(x,P)+\operatorname{cost}_2(y,P)]+[\operatorname{cost}_2(x,Q)-\operatorname{cost}_2(y,Q)]+\operatorname{cost}_2(y,P) \\
&\qquad (Q \text{ is } \tfrac{\epsilon}{2}\text{-RCDA}_2 \text{ of } P) \\
&\le \tfrac{\epsilon}{2}\operatorname{cost}_2(x,P)+(1+\tfrac{\epsilon}{2})\operatorname{cost}_2(y,P)+\epsilon\operatorname{cost}_2(y,Q) \\
&\qquad (\operatorname{cost}_2(x,Q)\le(1+\epsilon)\operatorname{cost}_2(x,Q)) \\
&\le \tfrac{\epsilon}{2}\operatorname{cost}_2(x,P)+(1+\tfrac{\epsilon}{2}+12\epsilon)\operatorname{cost}_2(y,P) \\
&\qquad (\text{by Proposition 3.4, } \operatorname{cost}_2(y,Q)\le 12\operatorname{cost}_2(y,P)) \\
&\le (1+\epsilon)(1+12.5\epsilon)\operatorname{cost}_2(y,P)\le(1+17\epsilon)\operatorname{cost}_2(y,P).
\end{aligned}$$

This establishes that Q is a stable $(\epsilon, 17\epsilon)$-coreset of P. ◀

References

1 Marcel R. Ackermann, Johannes Blömer, and Christian Sohler. Clustering for metric and nonmetric distance measures. *ACM Trans. Algorithms*, 6(4):59:1–59:26, 2010. doi:10.1145/1824777.1824779.

2 Peyman Afshani and Chris Schwiegelshohn. Optimal coresets for low-dimensional geometric median. In *ICML*, Proceedings of Machine Learning Research, pages 262–270. PMLR / OpenReview.net, 2024. URL: https://proceedings.mlr.press/v235/afshani24a.html.

3 Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi R. Varadarajan. Geometric approximation via coresets. In *Combinatorial and computational geometry*, volume 52 of *MSRI Publications*, chapter 1, pages 1–30. Cambridge University Press, 2005. URL: https://library2.msri.org/books/Book52/contents.html.

4 Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations*. Cambridge University Press, 2009.

5 Beatrice Bertolotti, Matteo Russo, Chris Schwiegelshohn, and Sudarshan Shyam. Simple and optimal sublinear algorithms for mean estimation. In *NeurIPS*, 2025. URL: https://openreview.net/forum?id=9MLfWJZXTn.

6 Johannes Blömer, Sascha Brauer, and Kathrin Bujna. Coresets for fuzzy k-means with applications. In *ISAAC*, volume 123 of *LIPIcs*, pages 46:1–46:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ISAAC.2018.46.

7 Vladimir Braverman, Vincent Cohen-Addad, Shaofeng H.-C. Jiang, Robert Krauthgamer, Chris Schwiegelshohn, Mads Bech Toftrup, and Xuan Wu. The power of uniform sampling for coresets. In *FOCS*, pages 462–473. IEEE, 2022. doi:10.1109/FOCS54457.2022.00051.

8 Vladimir Braverman, Shaofeng H.-C. Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for ordered weighted clustering. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 744–753. PMLR, 2019. URL: http://proceedings.mlr.press/v97/braverman19a.html.

9 Vladimir Braverman, Shaofeng H.-C. Jiang, Robert Krauthgamer, and Xuan Wu. Coresets for clustering in excluded-minor graphs and beyond. In *SODA*, pages 2679–2696. SIAM, 2021. doi:10.1137/1.9781611976465.159.

10 Mihai Bădoiu and Kenneth L. Clarkson. Optimal core-sets for balls. *Comput. Geom.*, 40(1):14–22, 2008. doi:10.1016/j.comgeo.2007.04.002.

11 Mihai Bădoiu, Sariel Har-Peled, and Piotr Indyk. Approximate clustering via core-sets. In *STOC*, pages 250–257. ACM, 2002. doi:10.1145/509907.509947.

12 Amir Carmel, Chengzhi Guo, Shaofeng H.-C. Jiang, and Robert Krauthgamer. Coresets for 1-center in ℓ_1 metrics. In *ITCS*, volume 325 of *LIPIcs*, pages 28:1–28:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.ITCS.2025.28.

13 Amir Carmel and Robert Krauthgamer. Stable coresets: Unleashing the power of uniform sampling. In *14th International Conference on Learning Representations, ICLR 2026*, 2026. URL: https://openreview.net/forum?id=sOpAa8iR0A.

14 Ching-Lueh Chang. On ultrametric 1-median selection. *Theoretical Computer Science*, 828-829:65–69, 2020. doi:10.1016/j.tcs.2020.04.021.

15 Ke Chen. On coresets for k-median and k-means clustering in metric and Euclidean spaces and their applications. *SIAM J. Comput.*, 39(3):923–947, 2009. doi:10.1137/070699007.

16 Xue Chen and Michal Derezinski. Query complexity of least absolute deviation regression via robust uniform convergence. In *COLT*, Proceedings of Machine Learning Research, pages 1144–1179. PMLR, 2021. URL: http://proceedings.mlr.press/v134/chen21d.html.

17 Herman Chernoff. *Sequential analysis and optimal design*. SIAM, 1972.

18 Rachit Chhaya, Anirban Dasgupta, Jayesh Choudhari, and Supratim Shit. On coresets for fair regression and individually fair clustering. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151, pages 9603–9625. PMLR, 2022. URL: https://proceedings.mlr.press/v151/chhaya22a.html.

19 Michael B. Cohen, Yin Tat Lee, Gary L. Miller, Jakub Pachocki, and Aaron Sidford. Geometric median in nearly linear time. In *STOC*, pages 9–21. ACM, 2016. doi:10.1145/2897518.2897647.

20 Vincent Cohen-Addad, Kasper Green Larsen, David Saulpic, and Chris Schwiegelshohn. Towards optimal lower bounds for k-median and k-means coresets. In *STOC*, pages 1038–1051. ACM, 2022. doi:10.1145/3519935.3519946.

21 Vincent Cohen-Addad, David Saulpic, and Chris Schwiegelshohn. Improved coresets and sublinear algorithms for power means in Euclidean spaces. In *NeurIPS*, pages 21085–21098. Curran Associates, Inc., 2021. URL: https://proceedings.neurips.cc/paper/2021/hash/b035d6563a2adac9f822940c145263ce-Abstract.html.

22 Vincent Cohen-Addad, David Saulpic, and Chris Schwiegelshohn. A new coreset framework for clustering. In *STOC*, pages 169–182. ACM, 2021. doi:10.1145/3406325.3451022.

23 Mónika Csikós and Nabil H. Mustafa. Optimal approximations made easy. *Inf. Process. Lett.*, 176:106250, 2022. doi:10.1016/J.IPL.2022.106250.

24 Ovidiu Daescu and Ka Yaw Teo. Fast deterministic approximation of medoid in r^{d}. In *CCCG*, pages 83–90, 2022.

25 Matan Danos. Coresets for clustering by uniform sampling and generalized rank aggregation. Master's thesis, Weizmann Institute of Science, Rehovot, Israel, 2021. URL: https://www.wisdom.weizmann.ac.il/~robi/files/MatanDanos-MScThesis-2021_11.pdf.

26 Michael Elkin, Arnold Filtser, and Ofer Neiman. Terminal embeddings. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX-/RANDOM*, volume 40 of *LIPIcs*, pages 242–264. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.242.

27 Ziyi Fang, Lingxiao Huang, and Runkai Yang. Coreset for robust geometric median: Eliminating size dependency on outliers. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025. URL: https://openreview.net/forum?id=Yhfx12Azz1.

28 Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *STOC*, pages 569–578. ACM, 2011. doi:10.1145/1993636.1993712.

29 Yehoram Gordon. Gaussian processes and almost spherical sections of convex bodies. *The Annals of Probability*, 16(1):180–188, 1988.

30 Sariel Har-Peled, Mitchell Jones, and Saladi Rahul. Active-learning a convex body in low dimensions. *Algorithmica*, 83(6):1885–1917, 2021. doi:10.1007/S00453-021-00807-W.

31 Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In *STOC*, pages 291–300. ACM, 2004. doi:10.1145/1007352.1007400.

32 Lingxiao Huang, Ruiyuan Huang, Zengfeng Huang, and Xuan Wu. On coresets for clustering in small dimensional euclidean spaces. In *ICML*, Proceedings of Machine Learning Research, pages 13891–13915. PMLR, 2023. URL: https://proceedings.mlr.press/v202/huang23h.html.

33 Lingxiao Huang, Shaofeng H.-C. Jiang, and Jianing Lou. The power of uniform sampling for k-median. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 13933–13956. PMLR, 2023. URL: https://proceedings.mlr.press/v202/huang23j.html.

34 Lingxiao Huang, Shaofeng H.-C. Jiang, Jianing Lou, and Xuan Wu. Near-optimal coresets for robust clustering. In *The Eleventh International Conference on Learning Representations, ICLR 2023*. OpenReview.net, 2023. URL: https://openreview.net/forum?id=Nc1ZkRW8Vde.

35 Lingxiao Huang, Shaofeng H.-C. Jiang, and Nisheeth K. Vishnoi. Coresets for clustering with fairness constraints. In *NeurIPS*, pages 7587–7598, 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/810dfbbebb17302018ae903e9cb7a483-Abstract.html.

36 Lingxiao Huang, Jian Li, Pinyan Lu, and Xuan Wu. Coresets for constrained clustering: General assignment constraints and improved size bounds. In *SODA*, pages 4732–4782. SIAM, 2025. doi:10.1137/1.9781611978322.161.

37 Lingxiao Huang, Jian Li, and Xuan Wu. On optimal coreset construction for Euclidean (k, z)-clustering. In *STOC*, pages 1594–1604. ACM, 2024. doi:10.1145/3618260.3649707.

38 Lingxiao Huang, K. Sudhir, and Nisheeth K. Vishnoi. Coresets for time series clustering. In *NeurIPS*, pages 22849–22862, 2021. URL: https://proceedings.neurips.cc/paper/2021/hash/c115ba9e04ab27fbbb664f932112246d-Abstract.html.

39 Lingxiao Huang and Nisheeth K. Vishnoi. Coresets for clustering in Euclidean spaces: importance sampling is nearly optimal. In *STOC*, pages 1416–1429. ACM, 2020. doi:10.1145/3357713.3384296.

40 Shaofeng H.-C. Jiang, Robert Krauthgamer, Jianing Lou, and Yubo Zhang. Coresets for kernel clustering. *Mach. Learn.*, 113(8):5891–5906, 2024. doi:10.1007/S10994-024-06540-Z.

41 Shaofeng H.-C. Jiang and Jianing Lou. Coresets for robust clustering via black-box reductions to vanilla case. In *ICALP*, volume 334 of *LIPIcs*, pages 101:1–101:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.ICALP.2025.101.

42 Yi Li, Philip M. Long, and Aravind Srinivasan. Improved bounds on the sample complexity of learning. *J. Comput. Syst. Sci.*, 62(3):516–527, 2001. doi:10.1006/jcss.2000.1741.

43 Shie Mannor and John N. Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *J. Mach. Learn. Res.*, 5:623–648, 2004. URL: https://jmlr.org/papers/volume5/mannor04b/mannor04b.pdf.

44 Stanislav Minsker, Sanvesh Srivastava, Lizhen Lin, and David B. Dunson. Scalable and robust bayesian inference via the median posterior. In *ICML*, JMLR Workshop and Conference Proceedings, pages 1656–1664. JMLR.org, 2014. URL: http://proceedings.mlr.press/v32/minsker14.html.

45 Alexander Munteanu and Chris Schwiegelshohn. Coresets-methods and history: A theoreticians design pattern for approximation and streaming algorithms. *Künstliche Intell.*, 32(1):37–53, 2018. doi:10.1007/S13218-017-0519-3.

46 James Newling and François Fleuret. A sub-quadratic exact medoid algorithm. In *AISTATS*, Proceedings of Machine Learning Research, pages 185–193. PMLR, 2017. URL: http://proceedings.mlr.press/v54/newling17a.html.

47 Andreas Nienkötter and Xiaoyi Jiang. Kernel-based generalized median computation for consensus learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(5):5872–5888, 2023. doi:10.1109/TPAMI.2022.3202565.

48 Aditya Parulekar, Advait Parulekar, and Eric Price. L1 regression with lewis weights subsampling. In *APPROX-RANDOM*, LIPIcs, pages 49:1–49:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.APPROX/RANDOM.2021.49.

49 Gideon Schechtman. A remark concerning the dependence on ϵ in Dvoretzky's theorem. In *Geometric Aspects of Functional Analysis: Israel Seminar (GAFA) 1987–88*, pages 274–277. Springer, 2006.

50 Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. Fair coresets and streaming algorithms for fair k-means. In *WAOA*, Lecture Notes in Computer Science, pages 232–251. Springer, 2019. doi:10.1007/978-3-030-39479-0_16.

51 S.A Shkarin. Isometric embedding of finite ultrametric spaces in banach spaces. *Topology and its Applications*, 142(1):13–17, 2004. doi:10.1016/j.topol.2003.12.002.

52 Christian Sohler and David P. Woodruff. Strong coresets for k-median and subspace approximation: Goodbye dimension. In *FOCS*, pages 802–813. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00081.

53 David P. Woodruff and Taisuke Yasuda. Coresets for multiple ℓ_p regression. In *ICML*, Proceedings of Machine Learning Research, pages 53202–53233. PMLR, 2024. URL: https://proceedings.mlr.press/v235/woodruff24a.html.

A Lower Bound

▶ **Theorem A.1.** *For every $\epsilon \in (0, 1/4)$ and every metric space $(\mathcal{X}, \mathrm{dist})$ containing at least two distinct points, there exists a finite point set $P \subseteq \mathcal{X}$, such that every algorithm producing a weak $(0, \epsilon)$-coreset (and in particular weak $(\epsilon/2, \epsilon)$-coreset) for 1-median with probability at least $4/5$, has query complexity $\Omega(\epsilon^{-2})$.*

Proof. Without loss of generality assume $\mathcal{X}$ contains two distinct points x, y of distance 1. For simplicity, assume $\epsilon/2$ divides n.

Consider two multisets P_0 and P_1. P_0 contains $n/2$ copies of x and $n/2$ copies of y, consequently, $\mathrm{cost}(x, P_0) = \mathrm{cost}(y, P_0) = 1/2$. P_1 contains $n(1-\epsilon)/2$ copies of x and $n(1+\epsilon)/2$ copies of y. For P_1, $\mathrm{cost}(x, P_1) = (1+\epsilon)/2$ and $\mathrm{cost}(y, P_1) = (1-\epsilon)/2$, so $\mathrm{cost}(x, P_1) > (1+2\epsilon)\,\mathrm{cost}(y, P_1)$.

Let Q be a multiset containing s copies of x and $m - s$ copies of y. Then $\mathrm{cost}(x, Q)/\,\mathrm{cost}(y, Q) = (m-s)/s$. If $s \geq m/2$, then $\mathrm{cost}(x, Q) \leq \mathrm{cost}(y, Q)$. Therefore, producing a weak coreset requires distinguishing whether Q was sampled from P_0 or P_1. As a weak $(0, \epsilon)$-coreset of P_1 must have y as an optimal solution. Under uniform sampling, $s \sim \mathrm{Binomial}(m, 1/2)$ for P_0 and $s \sim \mathrm{Binomial}(m, (1-\epsilon)/2)$ for P_1. By an information-theoretic lower bound, distinguishing between two binomial distributions whose biases differ by ϵ with constant probability requires $m = \Omega(\epsilon^{-2})$ samples [17, 43, 4]. ◀

Unbounded-Width CSPs Are Untestable in a Sublinear Number of Queries

Yumou Fei
Massachusetts Institute of Technology, Cambridge, MA, USA

Abstract

The bounded-degree query model, introduced by Goldreich and Ron (*Algorithmica, 2002*), is a standard framework in graph property testing and sublinear-time algorithms. Many properties studied in this model, such as bipartiteness and 3-colorability of graphs, can be expressed as satisfiability of constraint satisfaction problems (CSPs). We prove that for the entire class of *unbounded-width* CSPs, testing satisfiability requires $\Omega(n)$ queries in the bounded-degree model. This result unifies and generalizes several previous lower bounds. In particular, it applies to all CSPs that are known to be **NP**-hard to solve, including k-colorability of ℓ-uniform hypergraphs for any $k, \ell \geqslant 2$ with $(k, \ell) \neq (2, 2)$.

Our proof combines the techniques from Bogdanov, Obata, and Trevisan (*FOCS, 2002*), who established the first $\Omega(n)$ query lower bound for CSP testing in the bounded-degree model, with known results from universal algebra.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases constraint satisfaction problems, property testing

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.31

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2510.27012

Acknowledgements The author would like to thank Dor Minzer, Ronitt Rubinfeld, Shuo Wang and Standa Živný for many stimulating discussions during the development of this work. Special thanks to Ronitt Rubinfeld, for her helpful suggestions in improving the presentation of the paper.

1 Introduction

Property testing is a class of algorithmic problems in which the algorithm is allowed to inspect only a small portion of the input. In exchange for this limited access, the goal is relaxed: instead of deciding whether the input exactly satisfies a given property, the tester must merely distinguish inputs that satisfy the property from those that are *far* from satisfying it, under an appropriate notion of distance.

In this paper, we consider the problem of testing satisfiability of constraint satisfaction problems (CSPs). We formally define CSPs as follows. First of all, every CSP considered in this paper is specified by a finite *template*.

▶ **Definition 1.** *A* relational structure *is a pair (D, Γ) where D is a finite set, and Γ is a finite collection of relations on D. Each relation $R \in \Gamma$ is a function $R : D^k \to \{0, 1\}$ for some positive integer k (called the* arity *of R). A fixed relational structure (D, Γ) is also referred to as a* CSP template*.*

As a running example, if we let $D = \{0, 1\}$ and take $\Gamma = \{\mathbb{1}_{\{10,01\}}\}$, where $\mathbb{1}_{\{10,01\}}(x, y) = 1$ if and only if $x \neq y$, then the template (D, Γ) could express the bipartiteness property of graphs, as we will soon explain.


© Yumou Fei;
licensed under Creative Commons License CC-BY 4.0

Given a CSP template (D, Γ), we can collect all *instances* of the CSP into an infinite set $\mathrm{CSP}(\Gamma)$.

▶ **Definition 2.** *Given a CSP template (D, Γ), we define* $\mathrm{CSP}(\Gamma)$ *as the (infinite) collection of all* instances $I = (V, \mathcal{C})$*, where V is a finite set of* variables *and $\mathcal{C} = \{C_1, \ldots, C_m\}$ is a multi-set of* constraints *satisfying the following. Each constraint C_i is a pair $(\mathbf{v}, R)$, where $\mathbf{v} = (v_1, \ldots, v_k)$ is a tuple of distinct*[1] *variables from V, and $R : D^k \to \{0, 1\}$ is a relation from Γ. The variable set $\{v_1, \ldots, v_k\} \subseteq V$ is called the* scope *of C_i.*

Now an instance in $\mathrm{CSP}(\{\mathbb{1}_{\{10,01\}}\})$ can be viewed as a graph, with variables as vertices and constraints as edges. The graph is bipartite if and only if the CSP instance is "satisfiable," as defined in the following definition.

▶ **Definition 3.** *An* assignment *for an instance $I = (V, \mathcal{C}) \in \mathrm{CSP}(\Gamma)$ is a map $\tau : V \to D$. We say τ* satisfies *a constraint $((v_1, \ldots, v_k), R)$ if $R\big(\tau(v_1), \ldots, \tau(v_k)\big) = 1$. The* value *of an assignment τ, denoted by $\mathrm{val}_I(\tau)$, is the fraction of constraints in $\mathcal{C}$ satisfied by τ. Note that $\mathrm{val}_I(\tau) \in [0, 1]$. If $\mathrm{val}_I(\tau) = 1$, we say that τ* satisfies *the instance I, or τ is a* satisfying assignment *of I. The* value *of the instance I, denoted by val_I, is the maximum possible value any assignment may achieve.*

The CSP defined by the bipartiteness template $\big(\{0, 1\}, \{\mathbb{1}_{\{10,01\}}\}\big)$ will be denoted by 2COL. Similarly, the property of vertex 3-colorability in graphs can be expressed as a constraint satisfaction problem denoted by 3COL.

1.1 Testing CSP Satisfiability

We study the property testing of CSP satisfiability in the *bounded-degree query model*, defined in Definition 5 below. In this model, the input CSP instance has a fixed variable set, and the degree of each variable (i.e. the number of constraints involving the variable) in the instance is assumed to be uniformly bounded by a constant.

▶ **Definition 4.** *Let (D, Γ) be a CSP template and let V be a set of variables. We write $\mathrm{CSP}(\Gamma, V) \subseteq \mathrm{CSP}(\Gamma)$ for the collection of all $\mathrm{CSP}(\Gamma)$-instances whose variable set is V.*

▶ **Definition 5.** *Fix a CSP template (D, Γ). In the $\mathrm{BD}(d, n)$ model for testing satisfiability, the tester is given oracle access to a CSP instance $I = ([n], \mathcal{C}) \in \mathrm{CSP}(\Gamma, [n])$ with maximum degree at most d. The goal of an ε-tester, where $\varepsilon > 0$ is a fixed constant, is to achieve the following:*

1. *If I is satisfiable, the tester must accept I with probability at least $2/3$.*
2. *If at least εdn constraints must be removed from $\mathcal{C}$ to make I satisfiable, the tester must reject I with probability at least $2/3$.*

The tester accesses the instance I through oracle queries*: when the tester queries a variable $v \in [n]$, the oracle reveals an arbitrary one of the previously unseen constraints involving v, unless all constraints involving v have been revealed (in which case the oracle returns $\perp$).*

The bounded-degree query model was first introduced by [17], in the context of graph property testing. One of the earliest results in this model is that testing bipartiteness (i.e. testing satisfiability of 2COL instances) requires at least $\Omega(\sqrt{n})$ queries [17] (for some degree

[1] Some authors allow repetition of variables in a constraint. In our setting, one can also "artificially" allow repetitions by adding more relations to the relational structure. See the full version of this paper for details.

bound d and error $\varepsilon > 0$) and can be done with at most $\widetilde{O}(\sqrt{n})$ queries [16] (for any degree bound d and any error $\varepsilon > 0$). In contrast, Bogdanov, Obata, and Trevisan [6] showed that testing vertex 3-colorability requires $\Omega(n)$ queries.[2]

The study of testing CSP satisfiability in the bounded-degree model was already formally initiated in the early work of Bogdanov, Obata, and Trevisan [6], who showed that, beyond the specific case of 3COL, several other CSPs (such as 3SAT and 3LIN) require $\Omega(n)$ queries to test in this model. A more systematic investigation was later carried out by Yoshida [27], who proved that any CSP *robustly solved by BasicLP* (the basic linear program) has a satisfiability tester that makes a constant number of queries. Typical examples of such CSPs include 1SAT and 3HornSAT. For the satisfiability testing of any CSP that is not *solved by BasicLP*, [27] further showed that at least $\Omega(\sqrt{n})$ queries are necessary. Examples of such CSPs include 2SAT and 2COL.[3]

The main result of this paper strengthens this picture by proving that, among these BasicLP-unsolvable CSPs, the subclass of *unbounded-width* CSPs requires $\Omega(n)$ queries to test.

▶ **Theorem 6.** *For any CSP template (D, Γ) of unbounded width, there exist constants $\varepsilon \in (0, 1)$ and $d \in \mathbb{N}$ such that any ε-tester for satisfiability of $\mathrm{CSP}(\Gamma)$ instances in the $\mathrm{BD}(d, n)$ model must make $\Omega_{d,\varepsilon}(n)$ queries.*

The notion of bounded width (formally defined in Appendix A) originates from [12] and has been extensively studied in the universal-algebraic approach to CSPs. Bounded-width CSPs can be solved in polynomial time (see Proposition 15), so the class of unbounded-width CSPs includes all CSPs whose satisfiability is **NP**-hard to decide (unless $\mathbf{P} = \mathbf{NP}$, in which case all CSPs are trivially **NP**-hard). Consequently, Theorem 6 implies that all CSPs known to be **NP**-hard are also unconditionally (and maximally) hard to test in the bounded-degree model. This encompasses, for example, the property of k-colorability in ℓ-uniform hypergraphs for all $k, \ell \geqslant 2$ with $(k, \ell) \neq (2, 2)$ (see Remark A.5 of the full version). Our result thus unifies and extends several prior lower bounds from works such as [6, 28, 1].[4]

1.2 Discussion: Testing vs. Approximation

A property tester can be viewed as an algorithm that approximates the distance of an input to satisfying a property – albeit in a very coarse sense, as it only distinguishes zero distance from sufficiently large distance. Nevertheless, many property testers (for example, the constant-query tester of [27] for certain CSPs) operate by explicitly estimating this distance and checking whether it is close to zero. On the negative side, a hardness result for property testing (such as our Theorem 6) can often be interpreted as a hardness of approximation result. This connection to approximation algorithms was, in fact, one of the original motivations in [15], where graph property testing was first introduced.

In this subsection, we examine and discuss our main result Theorem 6 from the perspective of sublinear-time approximation algorithms.

---

[2] These seminal works, along with a large number of subsequent studies (such as [18, 5, 28, 29, 10], to name a few), also investigated a wide range of other graph properties within the bounded-degree model, many of which are not constraint satisfaction problems.

[3] Robust solvability by BasicLP is a stronger condition than solvability by BasicLP, so [27] left open the possibility of CSPs with superconstant but $o(\sqrt{n})$ query complexity. However, a recent unpublished note by Brady [7] proves that all CSPs solvable by BasicLP are also robustly solvable by it, thereby turning the result of [27] into a full dichotomy theorem.

[4] The recent work of [1] proves that testing k-colorability in k-uniform hypergraphs requires $\Omega(n)$ queries, for any $k \geqslant 3$.

1.2.1 MaxCSP Problems

A natural approximation problem associated with CSPs is that of estimating the value of a CSP instance (as defined in Definition 3).[5] This estimation task can be conveniently formulated in the following "gap version."

▶ **Definition 7.** *For a fixed CSP template (D, Γ), a completeness parameter $c \in (0, 1]$ and a soundness parameter $s \in [0, c)$, the problem $\mathsf{MaxCSP}(\Gamma)[c, s]$ is the promise problem where given an instance $I \in \mathrm{CSP}(\Gamma)$, the (randomized) algorithm should distinguish between the following two cases:*

1. *Yes case: if $\mathrm{val}_I \geqslant c$, then the algorithm should accept with probability at least $2/3$.*
2. *No case: if $\mathrm{val}_I \leqslant s$, then the algorithm should reject with probability at least $2/3$.*

The complexity of MaxCSP problems has been extensively studied in the context of polynomial-time algorithms, and a central question is to determine for which tuples (Γ, c, s) the problem $\mathsf{MaxCSP}(\Gamma)[c, s]$ can be solved in polynomial time. Notably, Raghavendra [24] almost[6] characterizes all such tuples, conditioned on the Unique Games Conjecture of [23].

1.2.2 MaxCSP vs. Satisfiability Testing

Whether property testing of CSP satisfiability has implications for MaxCSP problems depends on the notion of "distance to satisfiability" used by the property tester. In the d-bounded-degree model, having distance ε to satisfiability means that at least εdn constraints must be removed to make the instance satisfiable. In contrast, an instance with value $1 - \varepsilon$ in the MaxCSP sense requires removing at least an ε *fraction* of its constraints. Thus, testing CSP satisfiability in the bounded-degree model is analogous to solving $\mathsf{MaxCSP}(\Gamma)[1, 1 - \varepsilon]$, provided the CSP instance has $\Theta(n)$ constraints.

However, a CSP instance in the d-bounded-degree model $\mathrm{BD}(d, n)$ may have far fewer than $\Theta(n)$ constraints. In fact, if an instance contains only $O(1)$ constraints, it would be unreasonable to expect any sublinear-time algorithm to approximate its value, since with high probability the queried variables would not appear in any constraint. To address this, we introduce a modified model $\mathrm{BD}^*(d, \alpha, n)$, where $\alpha > 0$ is a fixed constant ensuring that all CSP instances considered contain at least αn constraints.

Under this refinement, Theorem 6 can be equivalently stated as follows:

▶ **Theorem 8.** *For any CSP template (D, Γ) of unbounded width, there exist constants $\varepsilon, \alpha \in (0, 1)$ and $d \in \mathbb{N}$ such that any algorithm for $\mathsf{MaxCSP}(\Gamma)[1, 1 - \varepsilon]$ in the $\mathrm{BD}^*(d, \alpha, n)$ model must make $\Omega(n)$ queries.*

It is easy to see that Theorem 8 implies Theorem 6. Due to the convenience provided by this rephrased version, in the rest of the paper we will work with Theorem 8 instead of Theorem 6.

---

[5] Ideally, an approximation algorithm would also produce an assignment achieving this value, i.e. one that satisfies as many constraints as possible. However, since we focus on sublinear-time algorithms in this paper, we cannot expect them to explicitly output a complete assignment.

[6] Except for the perfect-completeness case (i.e. the case $c = 1$), which remains a mystery.

1.2.3 Sublinear Time vs. Polynomial Time

As discussed in Section 1.2.2, because of the specific notion of "distance to satisfiability" used in the bounded-degree model, satisfiability testing in this setting most naturally corresponds to MaxCSP problems on instances with $\Theta(n)$ constraints. In contrast, the *dense graph model* (the more extensively studied model in graph property testing) uses another notion of distance, making it better aligned with MaxCSP problems on "dense" instances.

We argue that exactly due to this difference in instance regimes, the bounded-degree model provides a stronger connection to the study of polynomial-time algorithms for MaxCSPs, at least with respect to worst-case approximation ratios. First, MaxCSP problems on general instances can often be reduced to bounded-degree instances (see, e.g., [26]), so bounded-degree instances already capture much of the hardness in the problems. Second, MaxCSP problems tend to be easier on dense instances (see, e.g., [3]). Indeed, [2] shows that every CSP admits a constant-query satisfiability tester in the dense graph model. In contrast, in the bounded-degree model, satisfiability testing already exhibits a wide spectrum of complexities across CSP templates, with query complexity ranging from constant, to $\widetilde{\Theta}(\sqrt{n})$, up to $\Theta(n)$.

We next discuss some results about MaxCSP in the polynomial-time setting that are most relevant to this work.

In the context of polynomial-time algorithms, for certain CSP templates (D, Γ), the *perfect-completeness* problem $\mathsf{MaxCSP}(\Gamma)[1, 1-\varepsilon]$ can exhibit a markedly different level of complexity from the (potentially harder) *imperfect-completeness* problem $\mathsf{MaxCSP}(\Gamma)[1-\varepsilon', 1-\varepsilon]$, even when $\varepsilon' \to 0$ and ε is fixed. For example, in the case of $\mathsf{3LIN}$ (an unbounded-width CSP), the former problem admits a polynomial-time algorithm via Gaussian elimination, whereas the latter is **NP**-hard (due to [20]). More generally, Dalmau and Krokhin [11] proved that the imperfect-completeness problem is **NP**-hard for all unbounded-width CSP templates.

▶ **Theorem 9** ([11]). *For any CSP template (D, Γ) of unbounded width, there exists a constant $\varepsilon \in (0, 1)$ such that for every constant $\varepsilon' \in (0, \varepsilon)$, the problem $\mathsf{MaxCSP}(\Gamma)[1-\varepsilon', 1-\varepsilon]$ is* **NP***-hard.*

In terms of conclusions, our result is not directly comparable with Theorem 9, for multiple reasons. First, Theorem 9 is a hardness result for the imperfect completeness problem, while our Theorem 8 shows hardness for the (potentially easier) perfect completeness problem. Second, Theorem 9 is proved against polynomial-time algorithms, while Theorem 8 is against sublinear-query algorithms. Although **NP**-hardness is likely much stronger than a linear query lower bound, the latter does have the benefit of being unconditional. It is also not clear a priori whether some useful algorithms could have sublinear query complexity but super-polynomial time complexity, so **NP**-hardness usually does not immediately imply query lower bounds in sublinear-time models, even when assuming $\mathbf{P} \neq \mathbf{NP}$.

In terms of techniques, however, the proof of Theorem 8 runs in direct parallel to that of Theorem 9 in [11]. While [11] constructs a reduction from the **NP**-hardness of approximating $\mathsf{3LIN}$ instances (established by [20]), our proof of Theorem 8 builds an analogous reduction from the linear-query lower bound for $\mathsf{3LIN}$ instances (proved in [6]). Unlike in the **NP** setting, where a reduction need only run in polynomial time, our bounded-degree query setting requires addressing several additional subtleties. In particular, certain parts of the proof rely on a careful combination of techniques from both [6] and [11].

▶ Remark 10. The result of [11] (Theorem 9) was later shown to be tight by Barto and Kozik [4]. Specifically, they proved that for every CSP template (D, Γ) of bounded width and every constant $\varepsilon \in (0, 1)$, there exists a constant $\varepsilon' \in (0, \varepsilon)$ such that the problem

$\mathsf{MaxCSP}(\Gamma)[1-\varepsilon', 1-\varepsilon]$ belongs to $\mathbf{P}$. Together with the result of [11], this establishes that bounded width exactly characterizes the class of *robustly solvable* CSPs, as conjectured earlier by Guruswami and Zhou [19].

1.2.4 Sublinear Time vs. Sublinear Space

The past decades have seen a growing body of work on MaxCSP problems in *sublinear-space* algorithmic models (such as [21, 22, 9, 8, 25, 13], to name a few), particularly in the contexts of sketching and streaming algorithms. Recent studies [14, 13] reveal a phenomenological connection between the *multi-pass streaming* model and the bounded-degree query model: the class of MaxCSP problems that are easy in the bounded-degree model (admitting algorithms with constantly many queries) almost[7] coincides with those that are easy in the multi-pass streaming model (admitting algorithms with constantly many passes and polylogarithmic space). We view this connection as further motivation for studying MaxCSP and other approximation problems in the bounded-degree query model.

In light of this connection, a natural question arises: does the hardness result of Theorem 8 also extend to the multi-pass streaming model?

▶ **Question 11.** *Is it true that for every CSP template (D, Γ) of unbounded width, there exists a constant $\varepsilon > 0$ such that $\mathsf{MaxCSP}(\Gamma)[1, 1-\varepsilon]$ requires $\Omega(n)$ space in the multi-pass streaming model?*

We remark that, as discussed in [13], query lower bounds in sublinear time models are usually easier to establish than space lower bounds, and thus Theorem 8 can be seen as a preliminary step towards answering Question 11.

1.3 Open Problems

Perhaps the most obvious open problem left by this work is whether the result of Theorem 8 is tight, analogous to the case of Theorem 9 (see Remark 10).

▶ **Question 12.** *Do sublinear-query algorithms exist for satisfiability testing of all bounded-width CSPs?*

In their original work, Bogdanov, Obata and Trevisan [6] asked whether, given a CSP template (D, Γ), one can determine the optimal hardness gap ε for which $\mathsf{MaxCSP}(\Gamma)[1, 1-\varepsilon]$ requires $\Omega(n)$ queries. This question remains wide open (even for "simple" templates such as $\mathsf{3COL}$). Here, we pose the following more ambitious version.

▶ **Question 13.** *Determine the query complexity of $\mathsf{MaxCSP}(\Gamma)[c, s]$ in the bounded-degree model, for all predicate families Γ and parameters c and s.*

As discussed in Section 1.2.4, this question may also have strong relevance to sublinear-space computational models. Moreover, resolving Question 13 unconditionally (without assuming $\mathbf{P} \neq \mathbf{NP}$ or the Unique Games Conjecture) could possibly shed new light on the corresponding question for polynomial-time algorithms.

[7] Except for the perfect-completeness case, which remains a mystery for both models.

References

1 Hugo Aaronson, Gaia Carenini, and Atreyi Chanda. Property testing in bounded degree hypergraphs. *arXiv preprint arXiv:2502.18382*, 2025. doi:10.48550/arXiv.2502.18382.

2 Noga Alon and Asaf Shapira. Testing satisfiability. *Journal of Algorithms*, 47(2):87–103, 2003. doi:10.1016/S0196-6774(03)00019-1.

3 Sanjeev Arora, David Karger, and Marek Karpinski. Polynomial time approximation schemes for dense instances of np-hard problems. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 284–293, 1995. doi:10.1145/225058.225140.

4 Libor Barto and Marcin Kozik. Robustly solvable constraint satisfaction problems. *SIAM Journal on Computing*, 45(4):1646–1669, 2016. doi:10.1137/130915479.

5 Itai Benjamini, Oded Schramm, and Asaf Shapira. Every minor-closed property of sparse graphs is testable. *Advances in Mathematics*, 223:2200–2218, 2010.

6 Andrej Bogdanov, Kenji Obata, and Luca Trevisan. A lower bound for testing 3-colorability in bounded-degree graphs. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 93–102. IEEE, 2002. doi:10.1109/SFCS.2002.1181886.

7 Zarathustra Brady. Notes on csps and polymorphisms. *arXiv preprint arXiv:2210.07383*, 2022. doi:10.48550/arXiv.2210.07383.

8 Chi-Ning Chou, Alexander Golovnev, Madhu Sudan, Ameya Velingker, and Santhoshini Velusamy. Linear space streaming lower bounds for approximating csps. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 275–288, 2022. doi:10.1145/3519935.3519983.

9 Chi-Ning Chou, Alexander Golovnev, Madhu Sudan, and Santhoshini Velusamy. Sketching approximability of all finite csps. *Journal of the ACM*, 71(2):1–74, 2024. doi:10.1145/3649435.

10 Artur Czumaj, Morteza Monemizadeh, Krzysztof Onak, and Christian Sohler. Planar graphs: Random walks and bipartiteness testing. *Random Structures & Algorithms*, 55(1):104–124, 2019. doi:10.1002/RSA.20826.

11 Víctor Dalmau and Andrei Krokhin. Robust satisfiability for csps: Hardness and algorithmic results. *ACM Transactions on Computation Theory (TOCT)*, 5(4):1–25, 2013. doi:10.1145/2540090.

12 Tomás Feder and Moshe Y Vardi. The computational structure of monotone monadic snp and constraint satisfaction: A study through datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1998. doi:10.1137/S0097539794266766.

13 Yumou Fei, Dor Minzer, and Shuo Wang. A dichotomy theorem for multi-pass streaming csps. *arXiv preprint arXiv:2509.11399*, 2025. doi:10.48550/arXiv.2509.11399.

14 Yumou Fei, Dor Minzer, and Shuo Wang. Multi-pass streaming lower bounds for approximating max-cut. *arXiv preprint arXiv:2503.23404*, 2025. doi:10.48550/arXiv.2503.23404.

15 Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998. doi:10.1145/285055.285060.

16 Oded Goldreich and Dana Ron. A sublinear bipartiteness tester for bounded degree graphs. *Combinatorica*, 19(3):335–373, 1999. doi:10.1007/S004930050060.

17 Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2):302–343, 2002. doi:10.1007/S00453-001-0078-7.

18 Oded Goldreich and Dana Ron. On testing expansion in bounded-degree graphs. In *Studies in Complexity and Cryptography: Miscellanea on the Interplay between Randomness and Computation*, pages 68–75. Springer, 2011. doi:10.1007/978-3-642-22670-0_9.

19 Venkatesan Guruswami and Yuan Zhou. Tight bounds on the approximability of almost-satisfiable horn sat and exact hitting set. *Theory of Computing*, 8(1):239–267, 2012.

20 Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001. doi:10.1145/502090.502098.

21 Michael Kapralov, Sanjeev Khanna, and Madhu Sudan. Streaming lower bounds for approximating max-cut. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1263–1282. SIAM, 2014.

22 Michael Kapralov and Dmitry Krachun. An optimal space lower bound for approximating max-cut. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 277–288, 2019. doi:10.1145/3313276.3316364.

23 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775, 2002. doi:10.1145/509907.510017.

24 Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 245–254, 2008. doi:10.1145/1374376.1374414.

25 Raghuvansh R Saxena, Noah G Singer, Madhu Sudan, and Santhoshini Velusamy. Streaming algorithms via local algorithms for maximum directed cut. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3392–3408. SIAM, 2025. doi:10.1137/1.9781611978322.111.

26 Luca Trevisan. Non-approximability results for optimization problems on bounded degree instances. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 453–461, 2001. doi:10.1145/380752.380839.

27 Yuichi Yoshida. Optimal constant-time approximation algorithms and (unconditional) inapproximability results for every bounded-degree csp. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 665–674, 2011. doi:10.1145/1993636.1993725.

28 Yuichi Yoshida and Hiro Ito. Query-number preserving reductions and linear lower bounds for testing. *IEICE transactions on information and systems*, 93(2):233–240, 2010. doi:10.1587/TRANSINF.E93.D.233.

29 Yuichi Yoshida and Hiro Ito. Property testing on k-vertex-connectivity of graphs. *Algorithmica*, 62(3):701–712, 2012. doi:10.1007/S00453-010-9477-Y.

A Definition of Bounded Width

In this section, we provide the definition and basic properties of the "bounded width" notion for CSPs. The class of bounded-width CSP templates is defined by the correctness of an algorithm, which we call the width-(k, ℓ) algorithm, in deciding satisfiability of instances.

▶ **Definition 14** ([12]). *Let k, ℓ be integers such that $\ell \geqslant k \geqslant 1$. A CSP template (D, Γ) is said to have width (k, ℓ) if for all instances $I \in \mathrm{CSP}(\Gamma)$, Algorithm 1 correctly decides whether I is satisfiable. A template (D, Γ) is said to have bounded width if it has width (k, ℓ) for some integers k, ℓ; otherwise, we say (D, Γ) has unbounded width.*

Note that Algorithm 1 is always correct if it outputs `unsatisfiable`, so the only mistake is when the input instance is unsatisfiable but it outputs `satisfiable`. Also, since the running time of Algorithm 1 is clearly polynomial in $|V|$ and m, we immediately have the following property:

▶ **Proposition 15.** *For a CSP template (D, Γ), we let Γ-SAT be the problem of deciding satisfiability of instances from $\mathrm{CSP}(\Gamma)$. If (D, Γ) has bounded width, then Γ-SAT $\in \mathbf{P}$.*

Algorithm 1 Width-(k, ℓ) Algorithm.

Input : a CSP(Γ) instance $I = (V, (C_1, \ldots, C_m))$
Output : `satisfiable` or `unsatisfiable`

for *each subset* $U \subseteq V$ *of size* ℓ **do**
  Initialize a set of partial assignments $S_U \leftarrow D^U$
  for *each pair* (τ, i) *of partial assignment* $\tau \in D^U$ *and constraint index* $i \in [m]$ **do**
    if U *contains the scope of* C_i **and** τ *does not satisfy* C_i **then**
      $S_U \leftarrow S_U \setminus \{\tau\}$ `// remove inconsistencies with original constraints`

`// finish initialization; start main procedure`
repeat
  for *each pair of subsets* $U_1, U_2 \subseteq V$ *such that* $|U_1| = |U_2| = \ell$ **do**
    for *each (if any) subset* $W \subseteq U_1 \cap U_2$ *such that* $|W| = k$ **do**
      for *each* $\tau \in S_{U_1}$ **do**
        if $\tau_{|W} \neq \sigma_{|W}$ *for all* $\sigma \in S_{U_2}$ **then**
          $S_{U_1} \leftarrow S_{U_1} \setminus \{\tau\}$ `// remove inconsistencies in any` k `variables`
until *the sets* S_U *stop changing*
Output `unsatisfiable` if $S_U = \emptyset$ for some U; otherwise **output** `satisfiable`

Tight Simulation of a Distribution Using Conditional Samples

Tomer Adar
Technion – Israel Institute of Technology, Haifa, Israel

Abstract

We present an algorithm for simulating a distribution using prefix conditional samples (Adar, Fischer and Levi, 2024), as well as "prefix-compatible" conditional models such as the interval model (Cannone, Ron and Servedio, 2015) and the subcube model (CRS15, Bhattacharyya and Chakraborty, 2018). The sample complexity is $O(\log^2 N/\varepsilon^2)$ prefix conditional samples per query, which improves on the previously known $\tilde{O}(\log^3 N/\varepsilon^2)$ (Kumar, Meel and Pote, 2025). Moreover, our simulating distribution is $O(\varepsilon^2)$-close to the input distribution with respect to the Kullback-Leibler divergence, which is stricter than the usual guarantee of being $O(\varepsilon)$-close with respect to the total-variation distance.

We show that our algorithm is tight with respect to the highly-related task of estimation: every algorithm that is able to estimate the mass of individual elements within $(1 \pm \varepsilon)$-multiplicative error must make $\Omega(\log^2 N/\varepsilon^2)$ prefix conditional samples per element.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases Distribution learning, Distribution simulation, Probability estimation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.32

Category RANDOM

Related Version *Full Version*: https://arxiv.org/pdf/2506.18444

1 Introduction

Distribution testing, formally initiated by [4, 3], is a branch of property testing where the goal is distinguishing with high probability between distributions that have some property (for example, uniformity) and distributions that are far from any distribution with this property (for example, extremely biased distributions). "Farness" (or closeness) of distributions is usually determined by comparing the total-variation distance to a threshold parameter, usually denoted by ε.

The sampling model of distribution testing allows the algorithm to obtain a sequence of independent samples from its input distribution. One core result of distribution testing in the sampling model is uniformity testing over a domain of size N, at the cost of $\Theta(\sqrt{N}/\varepsilon^2)$ [15] samples. Other elementary properties, such as identity to an explicit distribution [3] and equivalence to an unknown distribution [11], require polynomial number of samples as well (but still sublinear).

The sampling model of distribution testing is ineffective for high-dimensional domains, such as the set of binary strings of length n, since it requires a polynomial number of samples in the domain size even for simple core properties, which is exponential in the dimension n. For obtaining sub-polynomial testing algorithms, we strengthen the model.

The conditional sampling model, independently introduced by [9] and [10], allows the algorithm to choose a set A and draw a sample from the input distribution when conditioned on belonging to A. Various works focus on restricted forms of the conditional sampling model, where only sets of a restricted form are eligible for conditioning. We recall a few of them.


© Tomer Adar;
licensed under Creative Commons License CC-BY 4.0



Subcube conditions

The subcube conditional model defined by [5] (see further works: [7, 12, 1, 13]) is defined for distributions over product sets: the domain is Σ^n for some alphabet Σ, and the sets eligible for condition are the cartesian product of their projections on individual characters. More formally, a subcube condition is a set of the form $A = \prod_{i=1}^{n} A_i$ for $A_1, \ldots, A_n \subseteq \Sigma$. Conceptually, the subcube conditional model resembles the "SELECT ... WHERE"-form of SQL queries.

Prefix conditions

In this paper, we focus on prefix conditions [1] (implicitly in [13, 5, 10]). Prefix conditions are a restricted form of subcube conditions in which there exists some k for which $A_1, \ldots, A_{k-1}$ are all singletons and $A_k, \ldots, A_n = \Sigma$. Note that we can identify every condition set $\{a_1\} \times \cdots \times \{a_{i-1}\} \times \Omega^{n-|w|}$ with the prefix $w = a_1 \ldots a_{i-1}$ (of length $i-1$) itself.

Additional works define various condition models such as *pair conditions* [9] (the condition set must have two elements), *interval conditions* [9] ($A = \{a, \ldots, b\} \subseteq \{1, \ldots, N\}$) and *coordinate conditions* [6] (we fix all symbols of a string but one). For binary string domains ($\Omega = \{0,1\}^n$), prefix conditions can be seen as prefix-interval condition sets (through the encoding $x \in \{0,1\}^n \to 1 + \sum_{i=1}^{n} 2^{n-i} x_i$ [1]) and coordinate conditions can be seen as subcube-pair condition sets.

1.1 Estimation and simulation of a distribution

In the string model of property testing (for example [16, 17]), the algorithm can choose an index to query and reveal the bit in that index. In other words, it obtains certain (undoubted) information about its input. In contrast, in all sampling models, the access to the input is indirect: the algorithm only obtains "behavioral" information about its input.

There are two concepts of "bridging" between sampling and querying. The first is *estimation*: given an element $x \in \Omega$, the algorithm estimates $\mu(x)$ (with high probability) within a reasonable error range using samples. The second is *simulation*: the algorithm constructs an explicit distribution $\tilde{\mu}$, which is hopefully close to the input distribution μ, and then queries it instead of the input distribution. Since approximating an entire distribution is too expensive, we allow a lazy construction of $\tilde{\mu}$, starting with "$\tilde{\mu}$ can be any distribution over Ω" and then gradually adding details required to answer queries.

Estimation and simulation are highly related tasks. While estimation demands "bounded worst-case error (for most input elements)", simulation of a close distribution can be seen as a "bounded expected estimation error (over the choice of the input element)". However, they are not fully compatible with each other:

- Assume that we can estimate all probabilities with a small error. Estimations can be biased, and therefore, the estimated probability masses do not necessarily sum to 1. Moreover, even if they do, the specific implementation of the estimator might be unable to *sample* the distribution implied by the individual estimations.
- Assume that we can access a distribution $\tilde{\mu}$ which is close to μ with respect to some divergence measure. For common divergences such as the KL-divergence, it might be the case that most elements are poorly estimated, but the estimation errors cancel each other, which results in a relatively low divergence.

One algorithmic paradigm which is compatible with both estimation and simulation is the distribution tree[1], or "slice and dice". First, we define a tree of sets eligible for conditioning by the model, where the root holds the whole domain Ω and every non-leaf node has two children whose disjoint union is the set it holds. Whenever an estimation $\mu(x)$ is required, we estimate the conditional mass $\mu(\mathsf{CHILD}|\mathsf{PARENT})$ in all nodes in the path from the root to the leaf containing the singleton $\{x\}$. To obtain the estimation, we use the chain rule: if $\{x\} = A_k \subseteq A_{k-1} \subseteq \cdots A_1 \subseteq A_0 = \Omega$, then $\Pr_\mu[x] = \prod_{i=1}^{k} \Pr_\mu[A_i|A_{i-1}]$. The estimation algorithm uses $\tilde{\mu}(x) = \prod_{i=1}^{k} \mathrm{est}(\Pr_\mu[A_i|A_{i-1}])$.

In [13] there is an explicit "slice-and-dice" algorithm for the subcube conditional sampling model. The algorithm can estimate $\mu(x)$ within $(1 \pm \varepsilon)$-factor for every x, unless one or more of its marginals is too close to 0 or to 1. The overall mass of these xs is bounded by $O(\varepsilon)$, which implies that the simulating distribution is $O(\varepsilon)$-close to μ with respect to the total-variation distance. The cost for every query is $\tilde{O}(n^3/\varepsilon^2)$ subcube conditional samples.

In this work we reduce the cost of the "slice and dice" method to $O(n^2/\varepsilon^2)$ conditional samples per query and improve the accuracy of the simulating distribution to be $O(\varepsilon^2)$-close with respect to the Kullback-Leibler divergence. Note that by Pinsker's inequality (see Lemma 59) it is never worse than $O(\varepsilon)$-total-variation-close.

The KL-divergence has two advantages over the total-variation distance for intermediate analysis: its propagation over multiple dimensions is tight (compare the equality in Lemma 62 to the corresponding upper bound for total-variation distance, for example in [5]), and it provides a refined divergence bound of indicator estimations (see Lemma 31 vs. the well-known $O(1/\sqrt{m})$-bound for total-variation distance), allowing for propagating over a higher dimension subject to the constraint $\mathrm{KL} = O(\mathrm{poly}(\varepsilon))$. We provide our algorithm in the prefix conditional model, but it can be implemented in every conditional model that resembles the "slice and dice" behavior (see Section 3 for more details).

Since explicitly learning a whole distribution is inevitably expensive (as we would have to store information about every element in the domain), we prefer to use local learning. Instead of storing a whole distribution, we provide an oracle access to the learned object, and store only the information needed to describe the queried parts. When processing a query, we compute only the (missing) parts needed to respond.

Notations

We refer to a *learning algorithm* whose input is an oracle access to a distribution μ, also called the *simulated distribution*, and its output is an oracle access to an output distribution $\tilde{\mu}$, which is called the *simulating distribution* (of μ).

1.2 Usage example

The distance estimation algorithms of [13, 2] consist of two modules: first, an estimation procedure $x \to \mathrm{est}(\mu(x))$, which we call a *mass query*, and second, an unconditional sampling algorithm that obtains the approximated mass of its samples and uses them to estimate the distance between the input distributions. While the first module requires a different logic for implementation in various models, the second module is fixed and stated in the following proposition.

[1] Note that a specific *implementation* inspired by this paradigm is not necessarily compatible with both.

▶ **Proposition 1** ([2], implicit in [13]). *Consider the following oracle for accessing a distribution μ: the oracle draws $x \sim \mu$, and returns a pair of the form $(x, (1 \pm O(\varepsilon))\mu(x))$. There exists an algorithm whose input is such an oracle access to two distributions μ and τ that makes $O(1/\varepsilon^2)$ oracle calls to estimate $d_{\mathrm{TV}}(\mu, \tau)$ within a $\pm\varepsilon$-additive error with probability at least $2/3$.*

1.3 Our results

The main result of this paper is the following theorem.

▶ **Theorem 2** (Informal statement of Theorem 33). *For every sufficiently small $\varepsilon > 0$ and sufficiently large $n \geq 1$, there exists an algorithm for simulating an input distribution over $\{0,1\}^n$ using $O(n^2/\varepsilon^2)$ prefix conditional samples per mass query and $O(n^2/\varepsilon^2)$ prefix conditional samples for sampling the simulating distribution. Moreover, the expected KL-divergence of the simulating distribution from the input distribution is $O(\varepsilon^2)$.*

Additionally, we show that any estimation algorithm must make $\Omega(n^2/\varepsilon^2)$ prefix conditional samples per query. The exact meaning of an "estimation algorithm" is detailed in the formal statement of Theorem 49.

▶ **Theorem 3** (Informal statement of Theorem 49). *Every estimation algorithm must make at least $\Omega(n^2/\varepsilon^2)$ prefix conditional samples per query.*

In Section 4, we provide a technical overview and the proof roadmap.

2 Preliminaries

In this section we formally define the notation scheme of this paper, as well as a few definitions more specific for this work. See Appendix B for formal definition of widely used distribution notations (such as Bernoulli) and for formal statements of well known facts (such as asymptotic bounds for $\ln x$ and Pinsker's inequality).

2.1 Basic notations

▶ **Definition 4** (Conditional distribution). *Let μ be a distribution over Ω and let $A \subseteq \Omega$ be a set. The* conditional distribution $\mu|^A$ *is defined as the distribution where $\Pr[x] = \frac{\Pr_\mu[x]}{\Pr_\mu[A]}$ for every $x \in A$ and $\Pr[x] = 0$ for every $x \notin A$. If $\Pr_\mu[A] = 0$, then the conditional distribution $\mu|^A$ is undefined.*

▶ **Definition 5** (Projected distribution, restricted distribution). *Let μ be a distribution over $\Omega = \prod_{i=1}^k \Omega_i$. For a non-empty set $I \subseteq \{1, \ldots, k\}$, the* projected distribution μ_I *is defined as the distribution over $\prod_{i \in I} \Omega_i$ for which $\Pr[x] = \Pr_{z \sim \mu}[\forall i \in I : z_i = x_i]$ for every $x \in \prod_{i \in I} \Omega_i$. In other words, it is the distribution obtained by drawing a sample from μ and* restricting *it to the entries in I's indexes.*

The overloaded notation $\mu|_I^A$ is interpreted as $(\mu|^A)_I$. In other words, we first sample according to the condition A, and then restrict the result to I's entries.

▶ **Definition 6** (Product of distributions). *Let $\mu_1, \ldots, \mu_k$ be distributions over $\Omega_1, \ldots, \Omega_k$, respectively. The* product distribution $\mu = \prod_{i=1}^k \mu_i$ *is the distribution over $\Omega = \prod_{i=1}^k \Omega_i$ where, for every $(a_1, \ldots, a_k) \in \Omega$, $\mu((a_1, \ldots, a_k)) = \prod_{i=1}^n \mu_i(a_i)$.*

▶ **Definition 7** (Total-variation distance). *Let μ and τ be two distributions over a domain Ω. Their* total-variation divergence *is $d_{\mathrm{TV}}(\mu,\tau) = \frac{1}{2}\sum_{x\in\Omega}|\mu(x)-\tau(x)|$.*

▶ **Definition 8** (Kullback-Leibler divergence). *Let μ and τ be two distributions over a domain Ω. The* KL-divergence *of μ from τ is $D_{\mathrm{KL}}(\mu \parallel \tau) = \mathrm{E}_{x\sim\mu}[\log_2(\mu(x)/\tau(x))]$.*

For convenience, we abuse the notation to define KL-divergence of probabilities:

▶ **Definition 9** (KL-divergence of probabilities (overloaded notation)). *For $p,q\in[0,1]$, we use the notation $D_{\mathrm{KL}}(p,q) = D_{\mathrm{KL}}(\mathrm{Ber}(p) \parallel \mathrm{Ber}(q)) = p\log\frac{p}{q} + (1-p)\log\frac{1-p}{1-q}$.*

2.2 Distribution testing models

A testing algorithm must access its input using oracles. Below we define two oracles for distribution testing.

▶ **Definition 10** (The sampling oracle). *Given an unknown input distribution μ over a set Ω, a call to the oracle results in a sample $x\in\Omega$ that distributes according to μ, independently of past calls to the oracle.*

▶ **Definition 11** (The conditional sampling oracle). *Given an unknown input distribution μ over a set Ω, the input of the oracle is a non-empty explicit subset $A\subseteq\Omega$ and the output is a sample $x\in\Omega$ that distributes like $\mu|^A$. If the mass of A in μ is zero, then the behavior of the oracle is undefined (it is only guaranteed that $x\in A$ with probability 1).*

▶ **Definition 12** (Prefix condition sets). *Let $\Omega=\Sigma^n$ be a domain. A set A is a* prefix condition *if there exists a string $w\in\bigcup_{i=0}^{n}\Sigma^i$ for which $A=\prod_{i=1}^{|w|}\{w_i\}\times\Sigma^{n-|w|}$.*

2.3 Algorithmic notions

In this subsection we formally define the notions of estimation and simulation of distributions.

Estimation algorithm must be able to correctly estimate the mass of *most* elements within a given *multiplicative factor*.

▶ **Definition 13** ((ε,c)-estimation algorithm). *Consider an algorithm whose input is a pair (μ,x), where μ is a distribution and x is an element in the domain of μ. For given parameters $0<\varepsilon<1$ and $0<c<1$, such an algorithm is an* (ε,c)-estimation algorithm *if for every input distribution μ there exists a set G_μ for which $\mu(G_\mu)\geq 1-c$ such that, for every $x\in G_\mu$, with probability at least $2/3$, the output of the algorithm for the input (μ,x) is a number in the range $(1\pm\varepsilon)\mu(x)$.*

A simulation algorithm must be defined through an interaction interface rather than a one-time call. It consists of a one-time initialization and a sequence of allowed interactions, "sample" and "query". After each interaction, the internal state of the simulation can mutate (in particular, the simulation can track its interaction history).

▶ **Definition 14** (Distribution simulation interface). *For a domain Ω, the interface of a* distribution simulation *consists of three procedures:*

- *Initialization: the input is a distribution μ and any additional parameters.*
- *Query: the input is an element $x\in\Omega$, and the output is a number $p\in[0,1]$.*
- *Sample: the output is an element $x\in\Omega$ and a number $p\in[0,1]$.*

▶ **Definition 15** (Interaction notation). *We formalize the q-round interaction of the caller with the simulation algorithm (Definition 14) as:*

- *A* request *sequence* $(\text{init}(\mu); X_1, \ldots, X_q)$*, where* $X_i \in \Omega \cup \{\perp\}$ *represents a query request (if* $X_i \in \Omega$*) or a sample request (*$X_i = \perp$*).*
- *A* response *sequence* $((\hat{X}_i, \hat{P}_i))_{1 \le i \le q}$*, where* $\hat{X}_i$ *represents the sampled element (for a sample request) or the queried element* X_i *(for a query request), and* $\hat{P}_i$ *represents the p-output.*

The simulation interface lacks of constraints ensuring that the output "looks like a simulation", which we have to formalize separately. Ideally, an implementation of the simulation interface secretly chooses a distribution $\tilde{\mu}$ (which is hopefully a good representation of μ) at the initialization phase, and then responds to sample and query interactions by sampling and querying the chosen $\tilde{\mu}$.

▶ **Definition 16** (The secret-distribution constraint). *An algorithmic implementation $\mathcal{A}$ for the distribution simulation interface follows the* secret-distribution constraint *if for every input distribution μ there exists a distribution $\mathcal{A}_\mu$ over distributions such that the interaction can be seen as drawing a secret distribution $\tilde{\mu} \sim \mathcal{A}_\mu$ in the initialization phase and then responding to sample and query requests through $\tilde{\mu}$. More precisely, given the request sequence $(\text{init}(\mu); X_1, \ldots, X_q)$, the probability to obtain the response $((\hat{X}_1, \hat{P}_1), \ldots, (\hat{X}_q, \hat{P}_q))$ is exactly:*

$$\sum_{\tilde{\mu}:\bigwedge_{i=1}^{q}(\tilde{\mu}(\hat{X}_i)=\hat{P}_i)} \left(\Pr_{\mathcal{A}_\mu}[\tilde{\mu}] \times \prod_{i=1}^{q} \begin{cases} 1 & X_i = \hat{X}_i \text{ (query)} \\ \tilde{\mu}(\hat{X}_i) & X_i = \perp \text{ (sample)} \\ 0 & X_i \neq \hat{X}_i, \perp \text{ (violation)} \end{cases} \right)$$

▶ **Definition 17** ((D, δ)-simulation of a distribution). *For a given divergence measure D and a parameter δ, an algorithmic implementation $\mathcal{A}$ of the distribution simulation interface is a* (D, δ)-simulation *if:*

- *Simulation behavior: $\mathcal{A}$ follows the secret-distribution constraint.*
- *Closeness: for every μ, regarding the distribution $\mathcal{A}_\mu$ (over distributions) required by the secret-distribution constraint,* $\mathrm{E}_{\tilde{\mu} \sim \mathcal{A}_\mu}[D(\tilde{\mu}|\mu)] \le \delta$.

3 Simulating conditional samples using prefix conditions

In this section we provide a conceptual reduction from conditional models compatible with the distribution-tree paradigm, which we formally define below, to the prefix conditional model.

First, we define a *breakdown tree*, which is the underlying structure of a distribution tree.

▶ **Definition 18** (A breakdown tree). *Let Ω be a domain. A* breakdown tree *is the following combinatorial tree structure:*

- *All leaves have the same depth.*
- *Every node holds a subset of the domain.*
- *A leaf can hold either the empty set or a singleton.*
- *The root holds the domain set Ω.*
- *Every non-leaf node has two children, arbitrarily named "left" and "right".*
- *Every non-leaf node holds the disjoint union of the subsets held by its children.*

▶ **Observation 19.** *Let $\mathcal{T}$ be a breakdown tree over Ω. For every $x \in \Omega$, there exists a single leaf that holds the singleton $\{x\}$.*

A conditional sampling model is *compatible with the distribution-tree paradigm* if for every set Ω which is valid as a domain of distributions in this model, there exists a breakdown tree in which every subset associated with a node is a valid condition set in the model. It is immediate to see that the subcube conditional model, the interval conditional model and the prefix conditional model are compatible with the distribution-tree paradigm, whereas the pair conditional model[2] is not.

We extend the structure of a breakdown tree to define distribution trees.

▶ **Definition 20** (A distribution tree). *Let Ω be a domain. A* distribution tree *consists of a breakdown tree and additionally:*

- *Every edge holds a probability.*
- *For every non-leaf node, the sum of its (two) outgoing edges is* 1.
- *The probability of an edge from a node holding a non-empty set to a child holding the empty set is* 0.
- *The probability of an edge from a node holding the empty set to its left child is* 1.

▶ **Definition 21** (Representation). *A distribution tree $\mathcal{T}$ of height ℓ over Ω* represents *a distribution μ over Ω if, for every $x \in \Omega$, considering the tree path $\Omega = A_0 \supseteq A_1 \supseteq \cdots \supseteq A_{\ell-1} \supseteq A_\ell = \{x\}$ and the weights $p_1, \ldots, p_\ell$ associated with the edges $A_0 \to A_1, \ldots, A_{\ell-1} \to A_\ell$, it holds that $\mu(x) = \prod_{i=1}^{\ell} p_i$.*

▶ **Observation 22.** *Let $\mathcal{T}$ be a breakdown tree over a domain Ω. There exists a bijection between distribution trees with the same structure of $\mathcal{T}$ and distributions over Ω such that every tree represents the distribution it is mapped to. Moreover, the mapped tree can be computed directly by associating every edge $A \to B$ ($A \supseteq B$) with the probability $\Pr_{z\sim\mu}[z \in B | z \in A]$.*

Observe that, given a breakdown tree of edge-height ℓ for a conditional model over a domain Ω, every condition associated with a node can be seen as a prefix condition in $\{0,1\}^\ell$, where the encoding of elements $x \in \Omega$ into elements in $\{0,1\}^\ell$ corresponds to the sequence of left and right edges in the path from the root to x.

Therefore, every prefix conditional model algorithm can be directly executed in every conditional model that has a breakdown tree, where the n parameter (for prefix algorithms over $\{0,1\}^n$) is the edge-height of the breakdown tree.

4 Overview

4.1 The prefix conditional model

We re-define the notions of a breakdown tree and a distribution tree to specifically match the behavior of prefix conditions.

▶ **Definition 23** (Prefix domain). *For $n \geq 1$, let $\{0,1\}^{<n} = \bigcup_{i=0}^{n-1} \{0,1\}^i$ be the set of all true prefixes of strings in $\{0,1\}^n$.*

Observe that there is a natural bijection between strings in $\{0,1\}^{<n}$ and non-singleton prefix conditions in $\{0,1\}^n$: a string $w \in \{0,1\}^{<n}$ corresponds to the prefix condition set $\{w\} \times \{0,1\}^{n-|w|}$. Moreover, the prefix conditional model has exactly one breakdown tree: the tree in which the non-leaf nodes are the sets corresponding to binary strings of length strictly less than n, the leaf nodes are the elements of $\{0,1\}^n$, and the children of a non-leaf node w correspond to the prefixes $w0$ and $w1$.

[2] The pair conditional model is defined in [9]. Beside the whole domain, the valid condition sets are all sets with exactly two elements.

▶ **Definition 24** (Marginal tree). *Let $\Omega = \{0,1\}^n$. A marginal tree consists of the (unique) breakdown tree and a function $\{0,1\}^{<n} \to [0,1]$ representing the probabilities associated with the one-edges (the edges of the form $w \to w1$).*

▶ **Observation 25** (Representation). *Given the marginal function $f : \{0,1\}^{<n} \to [0,1]$, the marginal tree defines a distribution in which, for every $x \in \{0,1\}^n$,*

$$\Pr[x] = \prod_{i=1}^{n}(x_i f(x_{1,\ldots,i-1}) + (1-x_i)(1-f(x_{1,\ldots,i-1}))) = \prod_{i=1}^{n}\begin{cases} f(x_{1,\ldots,i-1}) & x_i = 1 \\ 1-f(x_{1,\ldots,i-1}) & x_i = 0 \end{cases}$$

We use the specific structure of the only breakdown tree in the prefix conditional model to introduce a notation for marginal probabilities.

▶ **Definition 26** (Overloaded notation for marginals). *The marginal function $f : \{0,1\}^{<n} \to [0,1]$ represents a single distribution μ over $\{0,1\}^n$. For a prefix w of length $i-1$, we use the notation $\mu(x_i = 1|x_{1,\ldots,i-1} = w)$, its short form $\mu(x_i = 1|w)$ and its shorter form $\mu(1|w)$, to denote $f(w)$. Analogously, $\mu(x_i = 0|x_{1,\ldots,i-1} = w)$, $\mu(x_i = 0|w)$ and $\mu(0|w)$ denote $1 - f(w)$.*

In both the upper bound and the lower bound analysis, we use the chain rule for distributions over strings.

To *query* the mass of $x \in \{0,1\}^n$ with respect to μ, we multiply its marginal probabilities:

$$\mu(x) = \prod_{i=1}^{n}\mu(x_i|x_{1,\ldots,i-1}) = \prod_{i=1}^{n}(x_i f(x_{1,\ldots,i-1}) + (1-x_i)(1-f(x_{1,\ldots,i-1})))$$

To *estimate* $\mu(x)$, we multiply the estimations of its marginals.

$$\mathrm{est}(\mu(x)) = \prod_{i=1}^{n}\mathrm{est}(\mu(x_i|x_{1,\ldots,i-1})) = \prod_{i=1}^{n}(x_i\mathrm{est}(f(x_{1,\ldots,i-1})) + (1-x_i)(1-\mathrm{est}(f(x_{1,\ldots,i-1}))))$$

Note that the estimations of the marginals can be fully independent.

In parts of our analysis we use a weaker form of the prefix conditional sampling oracle.

▶ **Definition 27** (The marginal prefix sampling oracle). *Let μ be a distribution over $\Omega = \{0,1\}^n$. The input of a* marginal prefix sampling oracle *is a prefix condition $\{w\} \times \{0,1\}^{n-|w|}$ (for some $w \in \{0,1\}^{<n}$, and its output is a single bit that distributes like $\mu|_{|w|+1}^{w\times\{0,1\}^{n-|w|}}$.*

In other words, a marginal prefix oracle call draws a prefix conditional sample and then ignores all but the first free bit of the sample.

We can convert any prefix conditional sampling (including an unconditional sampling, in which the condition corresponds to the empty prefix) to a (random) sequence of marginal prefix samples whose length is the same as the number of free bits. Let $X_1, \ldots, X_n$ be the random variables for which the result sample is the concatenation $x = X_1 \cdots X_n$. Let $w \in \{0,1\}^{<n}$ of length $k-1$ ($1 \le k \le n$) be the prefix to condition on. For $1 \le i \le k-1$, X_i is the fixed value w_i. For $k \le i \le n$, once $X_1, \ldots, X_{i-1}$ are determined, we can sample X_i as the first bit of a sample conditioned on the prefix $X_1 \cdots X_{i-1}$.

4.2 The upper bound (Section 5)

We show that we can estimate a single marginal probability using m samples such that the expected KL-divergence of the estimation from the true marginal probability is bounded by $1/m$. By the chain rule, if we estimated all $2^n - 1$ marginal probabilities independently,

then the expected KL-divergence of the result distribution from the input distribution would be bounded by n/m. We choose $m = n/\varepsilon^2$ to have an $O(\varepsilon^2)$-D_{KL}-close distribution in expectation at the cost of $O(n/\varepsilon^2)$ prefix samples per marginal estimation. Since the marginal estimations are fully independent, given an element x we can estimate only its n marginals at the cost of $O(n^2/\varepsilon^2)$ prefix samples.

4.3 The lower bound (Section 6, Appendix A)

Recall that the input for the algorithm is pair of (1) a distribution over $\Omega = \{0,1\}^n$ and (2) an element $x \in \Omega$, and the output should be a number in the range $(1 \pm \varepsilon)\mu(x)$ with probability $2/3$.

We first describe the construction. We define two internal parameters, $\delta = \sqrt{2}\varepsilon/\sqrt{n}$ and $r = 8/\sqrt{n}$. For every $w \in \{0,1\}^{<n}$ we draw $s_w \in \{+1,-1\}$ uniformly and independently. We define the two distributions μ and μ' through its marginals. For every $w \in \{0,1\}^n$, we define:

- $\mu(1|w) = \frac{1}{2}(1 + s_w\delta)$, so that $\Pr_\mu[x] = 2^{-n}\prod_{i=1}^{n}(1 + (-1)^{1-x_i}s_{x_{1,\ldots,i-1}}\delta)$.
- $\mu'(1|w) = \frac{1}{2}(1 - s_w r)$, so that $\Pr_{\mu'}[x] = 2^{-n}\prod_{i=1}^{n}(1 + (-1)^{x_i}s_{x_{1,\ldots,i-1}}r)$.

We define two distributions over inputs, based on the choice of s_w variables:

- D_{yes}: the input provided to the algorithm is (μ, x), where x is drawn uniformly.
- D_{no}: the input provided to the algorithm is (μ, x), where x is drawn according to μ'.

The construction guarantees different concentration ranges for $\mu(x)$, as stated in the following lemma.

▶ **Lemma 28** (See Lemma 38, 43, 44). *For every sufficiently large $n \geq N_0$ and sufficiently small $0 < \varepsilon \leq \varepsilon_0$ there exist p_L and p_H for which:*

- $(1+\varepsilon)p_L < (1-\varepsilon)p_H$.
- *If (μ, x) is drawn from D_{yes}, then $\mu(x) \geq p_H$ with probability at least $1 - e^{-7.95}$.*
- *If (μ, x) is drawn from D_{no}, then $\mu(x) \leq p_L$ with probability at least $e^{-7.38}$.*

To prove a lower bound, we recall the meaning of a good estimation algorithm:

- The set of "good" elements has mass $1 - O(\varepsilon)$ according to the input distribution.
- Given a "good" element x, the output is in the range $(1 \pm \varepsilon)\mu(x)$ with probability at least $\frac{2}{3}$.

Note that the success probability in second requirement can be strengthen to $1 - c$ (instead of $2/3$) at the cost of $O(\log(1/c))$, which is asymptotically ignorable when c is a constant.

Consider an estimation algorithm $\mathcal{A}$ that can estimate $\mu(x)$ within a $(1 \pm \varepsilon)$-factor with probability at least $1 - e^{-20}$, unless x belongs to a small set "bad" elements whose mass is bounded by e^{-20}. In this setting,

$$\begin{aligned} d_{\mathrm{TV}}(\mathcal{A}(D_{\mathrm{yes}}), \mathcal{A}(D_{\mathrm{no}})) &\geq \Pr[\mathcal{A}(D_{\mathrm{yes}}) \geq (1-\varepsilon)p_H] - \Pr[\mathcal{A}(D_{\mathrm{yes}}) \geq (1-\varepsilon)p_H] \\ &\geq (e^{-7.38} - e^{-7.95}) - 4e^{-20} \geq e^{-8.25} \end{aligned}$$

Therefore, if an algorithm makes too few queries to distinguish between D_{yes} and D_{no} with total-variation distance at least $e^{-8.25}$, then it cannot be a good estimation algorithm. It remains to describe the lower bound for distinguishing between D_{yes} and D_{no} using prefix queries.

Let I_x be the set of prefixes (elements in $\{0,1\}^{<n}$) that are not prefixes of x. When x is explicitly given as an input, $\mu(x)$ is fully determined by the random sequence $(s_{x_{1,\ldots,i-1}})_{1 \leq i \leq n}$, which may depend on the choice of x. Also, note that if x is drawn uniformly, then $\mu(x)$ is independent of the assignment of all s_ws for $w \in I_x$. We observe that this independence is preserved even if x is drawn from μ' (Lemma 36).

Based on this analysis, we can define a random sequence $p_i = \mu(x_{1,\ldots,i}|x_{1,\ldots,i-1})$ (for $1 \le i \le n$).

- For D_{yes}, $p_i = (1+\delta)/2$ with probability $1/2$ and otherwise $p_i = (1-\delta)/2$.
- For D_{no}, $p_i = (1+\delta)/2$ with probability $(1-r)/2$ and otherwise $p_i = (1-\delta)/2$.

Clearly, distinguishing between D_{yes} and D_{no} cannot be easier than distinguishing between the sequences $(p_1, \ldots, p_n)$ drawn from D_{yes} and from D_{no}.

To show a lower bound for distinguishing between the p_i-sequence inputs, we use the following model that has been introduced in [14]: the input is a sequence of n coins, and in every step, the algorithm can choose a coin and sample it. Alternatively, the input can be seen as a sequence of inaccessible probabilities $p_1, \ldots, p_n \in [0,1]$, and in every step, the algorithm can choose an index $1 \le i \le n$ and sample a single bit that distributes like $\text{Ber}(p_i)$.

We use the following reduction for simulating the coin model using the prefix sampling model. The input of the distinguishing task corresponds to the sequence of marginals of a given random element x (that is, $p_1, \ldots, p_n$). Every prefix conditional sample translates into $O(1)$ steps in the coin model in expectation (see Lemma 48).

We can use the following result regarding a closely related task to deduce an $\Omega(1/r^2\delta^2) = \Omega(n^2/\varepsilon^2)$ lower bound. However, we explicitly prove hardness in Appendix A for completeness.

▶ **Lemma 29** (Rephrase of Theorem 3 in [14]). *Assume that for every $1 \le i \le n$, $p_i \ge (1+\delta)/2$ or $p_i \le (1-\delta)/2$. In this setting, estimating the fraction of positively-biased ($p_i \ge (1+\delta)/2$) within a $\pm r$-additive error requires $\Omega(1/r^2\delta^2)$ samples.*

4.4 Parameter table

Different sections use different parameter constraints. The upper bound uses n and ε. The lower bound for the sequence-distinguishing task uses n, δ and r, whereas the lower bound for estimation uses n and ε and explicitly defines δ, r.

Section	n	ε	δ	r
5 Upper bound	$n \ge 1$	$\varepsilon > 0$	N/A	N/A
6 Lower bound	$n \ge 4.5 \cdot 10^{16}$	$0 < \varepsilon \le e^{-11}$	$\sqrt{2}\varepsilon/\sqrt{n}$	$8/\sqrt{n}$
A Sequence distinguishing	by context	N/A	$0 < \delta < 1/3$	$0 \le r < 1/12$

5 The simulation algorithm

In this section we describe our simulation algorithm in the prefix conditional sampling model for an input distribution μ over $\Omega = \{0,1\}^n$.

Before we introduce our algorithm, we describe the common method (which is compatible with [13]). For every $1 \le i \le n$ and every $w \in \{0,1\}^{i-1}$, we use $O(n^2 \log(n/\varepsilon)/\varepsilon^2)$ samples to estimate the marginal probability $\mu(x_i = 1|w)$ within a $\left(1 \pm \frac{\varepsilon}{\sqrt{n}}\right)$-multiplicative factor with probability at least $1 - O(\varepsilon/n)$ (unless it is too close to zero or one, and in this case we estimate it as zero or one with probability $1 - O(\varepsilon/n)$).

Along the path from the root to a given leaf there are n edges, but errors of different directions (overshooting and undershooting) mostly cancel each other. Hence, with high probability, the accumulated error (when multiplying all n errors) in the range is $\left(1 \pm \frac{\varepsilon}{\sqrt{n}}\right)^{\pm\Theta(\sqrt{n})} = 1 \pm O(\varepsilon)$. By the union bound, the probability to obtain good estimations in all edges is at least $1 - O(\varepsilon)$. Overall, all elements have their mass estimated within

$(1 \pm O(\varepsilon))$-multiplicative error, except for a set of elements whose expected mass is $O(\varepsilon)$ as well. This implies that the expected total-variation distance of μ and the simulating distribution is $O(\varepsilon)$. The cost of querying the simulating distribution is $O(n^3 \log(n/\varepsilon)/\varepsilon^2)$, since we have to query n edges at the cost of $O(n^2 \log(n/\varepsilon)/\varepsilon^2)$ prefix samples per edge.

Our simulation algorithm uses the chain rule for D_{KL} (Lemma 62). Similarly to [5, 1], repeatedly applying the chain rule results in a bitwise identity.

▶ **Lemma 30** (Repeated chain rule for D_{KL})**.** *Let μ and τ be two distributions over $\{0,1\}^n$. In this setting, $D_{\mathrm{KL}}(\mu \parallel \tau) = \sum_{i=1}^{n} \mathrm{E}_{w \sim \mu|_{1,\ldots,i-1}} \left[D_{\mathrm{KL}}\left(\mu|_i^{\{w\}\times\{0,1\}^{n-i}} \,\middle\|\, \tau|_i^{\{w\}\times\{0,1\}^{n-i}} \right) \right]$.*

While the total-variation distance of Bernoulli distributions is linear, Lemma 61 states that the KL-divergence has a χ^2-like behavior, which is sublinear in many cases.

The following lemma is the core of our algorithm. It states that the expected KL-divergence of an estimated probability mass is linear in the number of samples, rather than its square-root as in total-variation distance.

▶ **Lemma 31.** *Let $p \in [0,1]$ and $m \geq 1$. In this setting, $\mathrm{E}_{t \sim \mathrm{Bin}(m,p)} \left[D_{\mathrm{KL}}(\frac{t}{m}, p) \right] \leq \frac{1}{m}$.*

Proof. For $p = 0$ and $p = 1$, $\Pr[t = mp] = 1$, and hence $\mathrm{E}[D_{\mathrm{KL}}(\frac{t}{m}, p)] = 0 \leq \frac{1}{m}$. For the rest of the proof we assume that $0 < p < 1$. In particular, we can safely divide by p and by $1 - p$.

Let $\hat{p} = \frac{t}{m}$ and $\delta = \frac{\hat{p}}{p} - 1$ (so that $\hat{p} = (1+\delta)p$). By Lemma 61, $D_{\mathrm{KL}}(\hat{p}, p) \leq \frac{(\hat{p}-p)^2}{p(1-p)} = \frac{\delta^2 p}{1-p}$. Therefore,

$$\begin{aligned}
\mathop{\mathrm{E}}_{t}[D_{\mathrm{KL}}(t/m, p)] \leq \frac{p}{1-p} \cdot \mathop{\mathrm{E}}_{t}\left[\left(\frac{t/m}{p} - 1 \right)^2 \right] &= \frac{p}{(mp)^2(1-p)} \cdot \mathop{\mathrm{E}}_{t}\left[(t - \mathrm{E}[t])^2 \right] \\
&= \frac{1}{m^2 p(1-p)} \mathrm{Var}[t] \\
&= \frac{1}{m^2 p(1-p)} p(1-p)m = \frac{1}{m}
\end{aligned}$$

◀

We show two implementations of $(D_{\mathrm{KL}}, \delta)$-simulation. The first, shown in Algorithms 2, 3 and 4 as a conceptual introduction, is extremely inefficient, but it is quite easy to show its correctness as a learning algorithm. The second, in Algorithms 5, 7 and 8, is efficient, and we prove its correctness by coupling its behavior with the behavior of the first implementation.

Both implementations use a common procedure Estimate-edge (Algorithm 1) that estimates the probability mass of a single edge by drawing $m = \lceil n/\delta \rceil$ prefix conditional samples and averaging the first free bit.

The first implementation has the learn-then-read concept. We exhaustively build the distribution tree of a distribution $\tilde{\mu}$ simulating the input distribution μ, and then query $\tilde{\mu}$ at individual elements whenever needed.

The initialization procedure, Learn-distribution, estimates all right edges (of the form $w \to w1$) independently.

The query procedure Query-learned-distribution and the sample procedure Sample-learned-distribution just access the explicitly-written learned distribution $\tilde{\mu}$.

▶ **Lemma 32.** *The triplet (Learn-distribution, Query-learned-distribution, Sample-learned-distribution) is a $(D_{\mathrm{KL}}, \delta)$-simulation of its input distribution.*

The second implementation has the lazy-construction concept. Since the estimations of Algorithm 2 are independent, it suffices to evaluate an edge when it (or its sibling edge) lies on a path of a query for the first time.

The initialization procedure Initialize-simulation initializes an empty memorization object.

Algorithm 1: Procedure Estimate-edge$(n, \mu, \delta; w, b)$.

Input: μ is the distribution over $\Omega = \{0,1\}^n$.
Input: δ is the accuracy parameter.
Input: $w \to wb$ is the edge to estimate.
Output: An estimation of $\mu(b|w)$.

1. Let $m \leftarrow \lceil n/\delta \rceil$.
2. Draw $x_1, \ldots, x_m$ independently from μ, conditioned on $x_{1,\ldots,|w|} = w$.
3. Let k be the number of samples $x_1, \ldots, x_m$ in which the $|w|+1$st bit is b.
4. Return k/m.

Algorithm 2: Procedure Learn-distribution(n, μ, δ).

Input: μ is the distribution over $\Omega = \{0,1\}^n$ to simulate.
Input: δ is the accuracy parameter.
Output: The simulating distribution $\tilde{\mu}$.
Accuracy: $\mathrm{E}\left[D_{\mathrm{KL}}(\tilde{\mu} \parallel \mu)\right] \leq \delta$.

1. Initialize the marginal function $f : \{0,1\}^{<n} \to [0,1]$, with all entries initially undefined.
2. For $w \in \{0,1\}^{<n}$ in an arbitrary order:
   a. Run and write $f(w) \leftarrow$ Estimate-edge$(n, \mu, \delta; w, 1)$.
3. Let $\tilde{\mu}$ be the distribution represented by the marginal function f. (Def. 24, Obs. 25)
4. Return $\tilde{\mu}$.

Algorithm 3: Procedure Query-learned-distribution$(\tilde{\mu}, x)$.

Input: $\tilde{\mu}$ is the learned distribution returned by Learn-distribution.
Output: The probability mass of x according to $\tilde{\mu}$.

1. Return $\tilde{\mu}(x)$.

Algorithm 4: Procedure Sample-learned-distribution$(\tilde{\mu})$.

Input: $\tilde{\mu}$ is the learned distribution returned by Learn-distribution.
Output: A sample x drawn from the learned distribution, and its probability mass $\tilde{\mu}(x)$.

1. Draw x from $\tilde{\mu}$.
2. Return $(x, \tilde{\mu}(x))$.

Algorithm 5: Procedure Initialize-simulation(n, μ, δ).

Input: μ is the distribution over $\Omega = \{0,1\}^n$ to simulate.
Input: δ is the accuracy parameter.

1. Return $(n, \mu, \delta, hist)$, where $hist$ is an empty list.

The sample and query procedures internally call the Access-simulation-edge procedure for an edge-by-edge execution.

The Access-simulation-edge procedure allows a memorized estimation of an edge. If the requested edge has never been estimated, then the procedure estimates it using Estimate-edge and then memorizes the result for both that edge and its sibling edge.

Algorithm 6: Procedure Access-simulation-edge(sim, w, b).

Input: sim: the simulation object obtained by Initialize-simulation.
Input: $w \to wb$ is edge to access.
Side effects: The $hist$ component of sim may change.

1. Let $n, \mu, \delta, hist$ be the components of sim as a 4-tuple.
2. If $hist$ contains $((w, b), ans)$:
   a. Let $f \leftarrow ans$.
3. Else:
   a. Let $f \leftarrow$ Estimate-edge$(n, \mu, \delta; w, b)$.
   b. Add $((w, b), f)$, $((w, 1-b), 1-f)$ to $hist$.
4. Return f.

The query and sample procedures execute an edge-by-edge logic. The edges are estimated through Access-simulation-edge.

Algorithm 7: Procedure Query-simulation(sim, x).

Input: sim: the simulation object obtained by Initialize-simulation.
Output: The probability mass of x according to the secret distribution.
Side effects: The $hist$ component of sim may change.

1. Let $n, \mu, \delta, hist$ be the components of sim as a 4-tuple.
2. Set $p_x \leftarrow 1$.
3. For i from 1 to n:
   a. Let $f_{x,i} \leftarrow$ Access-simulation-edge$(sim, x_{1,\ldots,i-1}, x_i)$.
   b. Set $p_x \leftarrow p_x \cdot f_{x,i}$.
4. Return p_x.

Algorithm 8: Procedure Sample-simulation(sim).

Input: sim: the simulation object obtained by Initialize-simulation.
Output: A sample x drawn from the secret distribution, and its probability mass according to that distribution.
Side effects: The *hist* component of sim may change.

1. Let $n, \mu, \delta, hist$ be the components of sim as a 4-tuple.
2. Set $p \leftarrow 1$.
3. Set $x \leftarrow$ the empty word.
4. For i from 1 to n:
   a. Let $f_i \leftarrow$ Access-simulation-edge$(sim, x, 1)$.
   b. Draw $b_i \in \{0, 1\}$ according to $\mathrm{Ber}(f_i)$.
   c. Set $x \leftarrow xb_i$. Append b to the end of x.
   d. If $b_i = 1$:
      i. Set $p \leftarrow p \cdot f_i$.
   e. Else:
      i. Set $p \leftarrow p \cdot (1 - f_i)$.
5. Return (x, p).

▶ **Theorem 33.** *The triplet (Initialize-simulation, Query-simulation, Sample-simulation) has the exact behavior (with respect to the user) as the triplet (Learn-distribution, Query-learned-distribution, Sample-learned-distribution). In particular, it is a $(D_{\mathrm{KL}}, \delta)$-simulation of its input distribution.*

Proof. The interaction of Query-simulation and Sample-simulation with the simulation object is only done through Access-simulation-edge calls. Since in Algorithm 2 the edges are independent, the specific order of assigning their weight is irrelevant. Each call to Access-simulation-edge assigns a weight to two sibling edges, independently of other edges. Therefore, the sequential access to the constructed distribution tree through Access-simulation-edge is identically distributed as evaluating all edges in advance as in Learn-distribution followed by reading the preprocessed data. ◀

We improve on [13] by applying Theorem 33 to Proposition 1.

▶ **Corollary 34.** *Let μ and τ be two distributions over $\Omega = \{0, 1\}^n$. Estimating the total-variation distance of μ and τ up to a $\pm\varepsilon$-additive error with probability at least $2/3$ can be done using $O(n^2/\varepsilon^4)$ prefix conditional samples.*

Proof. We use Theorem 33 to simulate μ and τ using random distributions $\tilde{\mu}$ and $\tilde{\tau}$ such that $\mathrm{E}[D_{\mathrm{KL}}(\tilde{\mu} \parallel \mu)] < \frac{1}{36}\varepsilon^2$ and $\mathrm{E}[D_{\mathrm{KL}}(\tilde{\tau} \parallel \tau)] < \frac{1}{36}\varepsilon^2$. After we initialize the simulation at no sample cost, we run a $O(1/\varepsilon^2)$-test of Proposition 1 to estimate the total-variation distance between $\tilde{\mu}$ and $\tilde{\tau}$ within $\pm\frac{1}{3}\varepsilon$-additive error. Such a test appears in [13] (inlined in a bigger subroutine) and in [2] (explicitly).

The complexity of the test is $O(1/\varepsilon^2)$ samples of the simulating distributions, each costing $O(n^2/\varepsilon^2)$ prefix conditional samples of the input distributions.

By Markov's inequality, with probability at least $(7/8)^2$, both $D_{\mathrm{KL}}(\tilde{\mu} \parallel \mu)$ and $D_{\mathrm{KL}}(\tilde{\tau} \parallel \tau)$ are at most $\frac{2}{9}\varepsilon^2$, and in this case, $|d_{\mathrm{TV}}(\mu, \tau) - d_{\mathrm{TV}}(\tilde{\mu}, \tilde{\tau})| \le d_{\mathrm{TV}}(\tilde{\mu}, \mu) + d_{\mathrm{TV}}(\tilde{\tau}, \tau) \le 2 \cdot \frac{1}{3}\varepsilon$.

Hence, estimating $d_{\mathrm{TV}}(\tilde{\mu},\tilde{\tau})$ within $\pm\frac{1}{3}\varepsilon$-additive error suffices to deduce a $\pm\varepsilon$-additive error estimation of $d_{\mathrm{TV}}(\mu,\tau)$. If we use $O(1)$ trials to amplify the success probability of estimating $d_{\mathrm{TV}}(\tilde{\mu},\tilde{\tau})$ to 8/9, then the success probability is at least $(7/8)^2 \cdot (8/9) > 2/3$. ◀

In particular, estimating the total-variation distance between two distributions over binary strings using subcube conditional sampling requires $O(n^2/\varepsilon^4)$ samples, a polynomial gain over the previously best known distance estimation algorithm for subcube conditions (by [13]), whose complexity is $\tilde{O}(n^3/\varepsilon^5)$ subcube conditional samples.

6 Lower bound

We show that estimation of the probability mass of individual elements using prefix conditional samples cannot be more efficient than simulating the input distribution as a whole (through Algorithms 5, 7, 8 and Theorem 33). Recall that we can fully define a distribution by assigning weights to the edges of the breakdown tree implied by the prefix condition sets.

▶ **Definition 35** (The counter-estimation distributions over inputs). *Let $\Omega = \{0,1\}^n$ be the domain and $\delta = \sqrt{2}\varepsilon/\sqrt{n}$, $r = 8/\sqrt{n}$ be internal parameters. For every $w \in \{0,1\}^{<n}$, we draw $s_w \in \{+1,-1\}$ uniformly and independently. We define the input distribution μ, and an additional distribution μ', through their marginals. For every $w \in \{0,1\}^{<n}$, we define $\mu(x_{|w|+1} = 1|w) = \frac{1+s_w\delta}{2}$ and $\mu'(x_{|w|+1} = 1|w) = \frac{1-s_w r}{2}$. In D_{yes}, we draw x uniformly and then return the pair (μ,x). In D_{no}, we draw $x \sim \mu'$ and return the pair (μ,x).*

We insist that μ is the input distribution in both cases. We only use μ' for drawing x in D_{no}.

▶ **Lemma 36.** *For every $x \in \{0,1\}^n$, let I_x be the set of prefixes $w \in \{0,1\}^{<n}$ that are not prefixes of x. In both D_{yes} and in D_{no}, when conditioned on an explicitly given x, the random variables $\{s_w\}_{w\in I_x}$ are drawn uniformly and independently from $\{+1,-1\}$.*

6.1 Concentration gap

For $n \geq 2$, $\varepsilon > 0$, $\delta = \sqrt{2}\varepsilon/\sqrt{n}$ and $r = 8/\sqrt{n}$, we define the following notations.

▶ **Definition 37** (A few technical definitions).

$$k_H = \frac{1}{2}n - \sqrt{3}\sqrt{n} \qquad k_L = \frac{1}{2}n - \sqrt{6}\sqrt{n}$$
$$p_H = 2^{-n}(1+\delta)^{k_H}(1-\delta)^{n-k_H} \qquad p_L = 2^{-n}(1+\delta)^{k_L}(1-\delta)^{n-k_L}$$
$$H_\mu = \{x : \mu(x) \geq p_H\} \qquad L_\mu = \{x : \mu(x) \leq p_L\}$$

The following lemmas (38-44) are technical.

▶ **Lemma 38.** *For $n \geq 1$ and $\varepsilon < 1/150$, $(1-\varepsilon)p_H > (1+\varepsilon)p_L$.*

▶ **Lemma 39.** *With probability 1 over the choice of μ, μ is ε-close to the uniform distribution over $\{0,1\}^n$.*

▶ **Lemma 40.** *For $n \geq 2.2 \cdot 10^8$, with probability 1 over the choice of μ, $\mu(H_\mu) \geq 1 - e^{-8}$.*

▶ **Lemma 41.** *For $n \geq 4.5 \cdot 10^{16}$ and $\varepsilon < \frac{1}{2000}$, with probability 1 over the choice of μ, $\mu(L_\mu) > e^{-14.57}$.*

▶ **Lemma 42.** *For $n \geq 3 \cdot 10^{10}$ and $\varepsilon > 0$, with probability 1 over the choice of μ and μ', for every $x \in L_\mu$, $\mu'(x) \geq e^{7.19}\mu(x)$.*

▶ **Lemma 43.** *For $n \geq 4.5 \cdot 10^{16}$ and $\varepsilon \leq e^{-11}$, $\Pr_{(\mu,x)\sim D_{\mathrm{yes}}}[\mu(x) \geq p_H] \geq 1 - e^{-7.95}$.*

▶ **Lemma 44.** *For $n \geq 4.5 \cdot 10^{16}$ and $\varepsilon \leq e^{-11}$, $\Pr_{(\mu,x)\sim D_{\mathrm{no}}}[\mu(x) \leq p_L] \geq e^{-7.38}$.*

6.2 The lower bound theorem

▶ **Lemma 45.** *For $n \geq 4.5 \cdot 10^{16}$ and $\varepsilon \leq e^{-11}$, let $\mathcal{A}$ be an algorithm that uses prefix conditional samples and results in a real number such that for every distribution μ over $\Omega = \{0,1\}^n$ there exists a set $G_\mu \subseteq \Omega$ of mass $\Pr_\mu[G_\mu] > 1 - e^{-20}$ for which $\Pr[\mathcal{A}(\mu, x) \in (1 \pm \varepsilon)\mu(x)] > 1 - e^{-20}$ for every $x \in G_\mu$. In this setting, the total variation distance of the behaviors of the algorithm when given an input drawn from D_{yes} and from D_{no} is greater than $e^{-8.25}$.*

Note that, for estimation algorithms, having $\mu(G_\mu) = 1$ is infeasible, and general upper bound algorithms usually guarantee that $\mu(G_\mu) = 1 - O(\varepsilon)$. Here we show a lower bound for a much weaker demand, that allows for excluding a fixed weight of the elements.

Proof. Consider the distributions D_{yes} and D_{no} and the following algorithm: we use $\mathcal{A}$ to estimate $(1 \pm \varepsilon)\mu(x)$. If the result estimation is greater than $(1 - \varepsilon)p_H$ then we accept, and otherwise we reject. By Lemma 38, the ranges $(1 \pm \varepsilon)p_H$ and $(1 \pm \varepsilon)p_L$ are disjoint, and therefore, a $1 \pm \varepsilon$-multiplicative error suffices to distinguish between p_H and p_L.

Let B_{wrong} be the bad event "the estimation of $\mu(x)$ is outside the range $(1 \pm \varepsilon)\mu(x)$". The probability to reject an input (μ, x) drawn from D_{yes} is bounded by:

$$\begin{aligned}
\Pr[\text{REJECT}|D_{\text{yes}}] &\leq \Pr_{D_{\text{yes}}}[x \notin H_\mu \cap G_\mu] + \Pr[B_{\text{wrong}}|x \in G_\mu] \\
&\leq d_{\text{TV}}(\mathcal{U}, \mu) + (1 - \mu(H_\mu)) + (1 - \mu(G_\mu)) + e^{-20} \\
\text{[L 39, L 40]} &\leq \varepsilon + e^{-8} + e^{-20} + e^{-20}
\end{aligned}$$

By Lemma 41 and Lemma 42,

$$\mu'(L_\mu \cap G_\mu) \geq \min_{x \in L_\mu} \frac{\mu'(x)}{\mu(x)} \cdot (\mu(L_\mu) - (1 - \mu(G_\mu))) \geq e^{7.19} \cdot (e^{-14.57} - e^{-20}) \geq e^{-7.385}$$

The probability to reject an input (μ, x) drawn from D_{no} is at least:

$$\Pr[\text{REJECT}|D_{\text{no}}] \geq \Pr_{D_{\text{no}}}[x \in G_\mu \cap L_\mu] \cdot \Pr[\neg B_{\text{wrong}}|x \in G_\mu] \geq e^{-7.385} \cdot (1 - e^{-20}) \geq e^{-7.39}$$

Therefore, the distance in behaviors of the algorithms when running on D_{yes} and D_{no} is at least $e^{-7.39} - (\varepsilon + e^{-8} + e^{-20} + e^{-20}) > e^{-8.25}$ (for $\varepsilon \leq e^{-11}$). ◀

▶ **Lemma 46.** *For $n \geq 4.5 \cdot 10^{16}$ and $\varepsilon < e^{-11}$, let $\mathcal{A}$ be an algorithm that uses marginal prefix conditional samples and results in a real number such that for every distribution μ over $\Omega = \{0,1\}^n$ there exists a set $G_\mu \subseteq \Omega$ of mass $\Pr_\mu[G_\mu] > 1 - e^{-20}$ for which $\Pr[\mathcal{A}(\mu, x) \in (1 \pm \varepsilon)\mu(x)] > 1 - e^{-20}$ for every $x \in G_\mu$. In this setting, $\mathcal{A}$ must make at least $\frac{n^2}{2.8 \cdot 10^{25} \varepsilon^2}$ marginal prefix conditional samples that contain x in expectation.*

Proof. Observe that, for every $w \in \{0,1\}^{<n}$, the marginal prefix conditional sample $\mu|_{|w|+1}^{w \times \{0,1\}^{n-|w|}}$ is identical to drawing a bit from $\text{Ber}\left(\frac{1+s_w\delta}{2}\right)$. By Lemma 36, the random variables $\{s_w\}_{w \in I_x}$, for $I_x = \{0,1\}^{<n} \setminus \{1 \leq i \leq n : x_{1,\ldots,i-1}\}$, are independent, and distribute the same in D_{yes} and in D_{no}. Hence, sampling them is redundant.

In other words, in every non-redundant step, the algorithm chooses some $1 \leq i \leq n$ and samples a bit whose probability to be 1 is $\frac{1+s_{x_{1,\ldots,i-1}}\delta}{2}$. For every $1 \leq i \leq n$, let $p_i = \frac{1+s_{x_{1,\ldots,i-1}}\delta}{2}$.

In D_{yes}, every s_w is uniformly drawn from $\{+1, -1\}$, and hence, the p_is are uniformly drawn from $\frac{1 \pm \delta}{2}$. This matches the $D_n(\delta, 0)$ distribution of the sequence-distinguishing task. In D_{no}, when conditioned on drawing x from μ', every s_w (for w on the path of x) is $+1$ with probability $\frac{1-r}{2}$ and otherwise -1. This matches the $D_n(\delta, r)$ distribution of the sequence-distinguishing task.

Note that this analysis is oblivious of the dependence between the s_ws and x. Observe that if we denote $s_{x_{1,\dots,i-1}}$ by $\hat{s}_{i-1}$, then the sequence $(\hat{s}_0,\dots,\hat{s}_{n-1})$ is independent of x and iid.

Let q be the expected number of non-redundant steps. By Markov's inequality, with probability at least $1-e^{-9}$ it is bounded by $e^9 q$.

If $e^9 q \le \frac{(e^{-9})^4}{4\cdot 10^5\delta^2}n = \frac{n^2}{e^{36}\cdot 4\cdot 10^5\cdot 2\varepsilon^2}$, then by Lemma 56, the total variation distance between the runs is less than e^{-9}.

Overall, if $q \le \frac{n^2}{2.8\cdot 10^{25}\varepsilon^2} \le \frac{n^2}{e^9\cdot e^{36}\cdot 4\cdot 10^5\cdot 2\varepsilon^2}$, then the total-variation distance between the run is bounded by $e^{-9}+e^{-9} < e^{-8.25}$. By Lemma 45, a "good" estimation algorithm behaves more differently ($d_{\mathrm{TV}} > e^{-8.25}$) on D_{yes} on D_{no}, and hence, every such an algorithm must make at least $\frac{n^2}{2.8\cdot 10^{25}\varepsilon^2}$ prefix conditional queries. ◀

We describe a setting for the following lemma.

▶ **Definition 47** (Setting for Lemma 48). *Let $x \in \{0,1\}^n$, $1 \le i \le n$ and $w \in \{0,1\}^{i-1}$ be a prefix. Let $X_i,\dots,X_n$ be the bits of a prefix conditional sample $\mu|^{w\times\{0,1\}^{n-i}}$. Let $W_{i-1},\dots,W_{n-1}$ be the intermediate prefixes ($W_{i-1}=w$, $W_{j-1}=W_{j-2}X_{j-1}$ for $i+1\le j\le n$). Also, let $N_{\mathrm{effective}} = |\{i\le j\le n : W_{j-1} \text{ is a prefix of } x\}|$.*

As observed before, the sampling from $\mu|^{w\times\{0,1\}^{n-i}}$ can be decomposed into $n-i+1$ marginal samplings such that for $i\le j\le n$, after W_{i-1} is determined, X_j distributes as $\mu|_j^{W_{j-1}\times\{0,1\}^{n-j}}$. Therefore, $N_{\mathrm{effective}}$ describes the number of effective marginal samples.

▶ **Lemma 48.** *In the setting of Definition 47, if $n \ge 20$, then $\mathrm{E}[N_{\mathrm{effective}}] \le 3$.*

▶ **Theorem 49.** *For $n \ge 4.5\cdot 10^{16}$ and $\varepsilon \le e^{-11}$, let $\mathcal{A}$ be an algorithm that uses prefix conditional samples and results in a real number such that for every distribution μ over $\Omega = \{0,1\}^n$ there exists a set $G_\mu \subseteq \Omega$ of mass $\Pr_\mu[G_\mu] > 1-e^{-20}$ for which $\Pr[\mathcal{A}(\mu,x)\in(1\pm\varepsilon)\mu(x)] > 1-e^{-20}$ for every $x\in G_\mu$. In this setting, $\mathcal{A}$ must use $\Omega(n^2/\varepsilon^2)$ prefix conditional samples.*

Proof. Recall that we can see every prefix condition sampling with k free bits as a (random) sequence of k marginal prefix condition sampling. In the ith step, we condition on the first already-fixed $n-k$ bits and on the additional $i-1$ already-sampled bits to obtain the $n-k+i$th bit. By Lemma 48, for every individual prefix sample, the expected number of marginal samples whose condition contains x is bounded by 3. By linearity of expectation, the expected number of marginal samples whose prefix condition contains x is bounded by $3q$.

By Lemma 46, since the expected number of "non-redundant" samples is $O(q)$, the algorithm must make $q=\Omega(n^2/\varepsilon^2)$ prefix conditional samples. ◀

References

1. Tomer Adar, Eldar Fischer, and Amit Levi. Improved bounds for high-dimensional equivalence and product testing using subcube queries. In *Approximation, Randomization, and Combinatorial Optimization Algorithms and Techniques (APPROX/RANDOM)*, 2024.
2. Tomer Adar, Eldar Fischer, and Amit Levi. Optimal mass estimation in the conditional sampling model. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2026.
3. Tugkan Batu, Eldar Fischer, Lance Fortnow, Ravi Kumar, Ronitt Rubinfeld, and Patrick White. Testing random variables for independence and identity. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pages 442–451. IEEE, 2001. doi:10.1109/SFCS.2001.959920.

4 Tugkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D Smith, and Patrick White. Testing that distributions are close. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 259–269. IEEE, 2000. doi:10.1109/SFCS.2000.892113.

5 Rishiraj Bhattacharyya and Sourav Chakraborty. Property testing of joint distributions using conditional samples. *ACM Transactions on Computation Theory (TOCT)*, 10(4):1–20, 2018. doi:10.1145/3241377.

6 Antonio Blanca, Zongchen Chen, Daniel Štefankovič, and Eric Vigoda. Complexity of high-dimensional identity testing with coordinate conditional sampling. *ACM Transactions on Algorithms*, 21(1):1–58, 2024. doi:10.1145/3686799.

7 Clément L Canonne, Xi Chen, Gautam Kamath, Amit Levi, and Erik Waingarten. Random restrictions of high dimensional distributions and uniformity testing with subcube conditioning. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 321–336. SIAM, 2021. doi:10.1137/1.9781611976465.21.

8 Clément L Canonne, Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. Testing bayesian networks. In *Conference on Learning Theory*, pages 370–448. PMLR, 2017. URL: http://proceedings.mlr.press/v65/canonne17a.html.

9 Clément L Canonne, Dana Ron, and Rocco A Servedio. Testing probability distributions using conditional samples. *SIAM Journal on Computing*, 44(3):540–616, 2015. doi:10.1137/130945508.

10 Sourav Chakraborty, Eldar Fischer, Yonatan Goldhirsh, and Arie Matsliah. On the power of conditional samples in distribution testing. *SIAM Journal on Computing*, 45(4):1261–1296, 2016. doi:10.1137/140964199.

11 Siu-On Chan, Ilias Diakonikolas, Paul Valiant, and Gregory Valiant. Optimal algorithms for testing closeness of discrete distributions. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1193–1203. SIAM, 2014. doi:10.1137/1.9781611973402.88.

12 Xi Chen and Cassandra Marcussen. Uniformity testing over hypergrids with subcube conditioning. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 4338–4370. SIAM, 2024. doi:10.1137/1.9781611977912.152.

13 Gunjan Kumar, Kuldeep S Meel, and Yash Pote. Distance estimation for high-dimensional discrete distributions. In *The 28th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2025.

14 Jasper CH Lee and Paul Valiant. Uncertainty about uncertainty: Optimal adaptive algorithms for estimating mixtures of unknown coins. *arXiv e-prints*, pages arXiv–1904, 2019.

15 Liam Paninski. A coincidence-based test for uniformity given very sparsely sampled discrete data. *IEEE Transactions on Information Theory*, 54(10):4750–4755, 2008. doi:10.1109/TIT.2008.928987.

16 Ronitt Rubinfeld and Madhu Sudan. Self-testing polynomial functions efficiently and over rational domains. In *Proceedings of the third annual ACM-SIAM symposium on Discrete algorithms*, pages 23–32, 1992. URL: http://dl.acm.org/citation.cfm?id=139404.139410.

17 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996. doi:10.1137/S0097539793255151.

18 Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 222–227, 1977.

A Lower bound for the sequence-distinguishing task

In this appendix we show a lower bound for the sequence-distinguishing task. The goal is distinguishing between the following random sequences, regarding the parameters $\delta > 0$ and $r \geq 0$:

- $D_n(\delta, 0)$: $p_i = (1+\delta)/2$ with probability $1/2$, and otherwise $p_i = (1-\delta)/2$, independently for every $1 \le i \le n$.
- $D_n(\delta, r)$: $p_i = (1+\delta)/2$ with probability $(1-r)/2$, and otherwise $p_i = (1-\delta)/2$, independently for every $1 \le i \le n$.

In each step, the algorithm can choose an index $1 \le i \le n$ and obtain a bit drawn from $\mathrm{Ber}(p_i)$.

We show that, if $n \le 64/r^2$, then we must make $\Omega(1/r^2\delta^2)$ queries, even if we allow adaptive behavior.

A.1 Technical lemmas

We first state a few technical lemmas.

▶ **Lemma 50.** *Let $0 \le t \le 1/4$. For two distributions μ and ν over Ω for which $\mu(x) \in (1 \pm t)\nu(x)$ for every $x \in \Omega$, $D_{\mathrm{KL}}(\mu \parallel \nu) \le t^2/\ln 2$.*

▶ **Lemma 51.** *Let μ and ν be two distributions over a domain Ω. It holds that:*

$$\sum_{x \in \Omega} \frac{(\mu(x) - \nu(x))^2}{\mu(x) + \nu(x)} \le (D_{\mathrm{KL}}(\mu \parallel \nu) + D_{\mathrm{KL}}(\nu \parallel \mu)) \cdot \ln 2$$

▶ **Lemma 52.** *Let μ and ν be two distributions over a domain Ω. If $|r| < \frac{1}{2}$, then:*

$$D_{\mathrm{KL}}\left(\frac{1}{2}\mu + \frac{1}{2}\nu \,\middle\|\, \frac{1+r}{2}\mu + \frac{1-r}{2}\nu\right) \le \frac{1}{2}r^2(D_{\mathrm{KL}}(\mu \parallel \nu) + D_{\mathrm{KL}}(\nu \parallel \mu))$$

A.2 Lower bound for non-adaptive algorithms

Before we introduce the lower bound against adaptive algorithms, we state a lower bound for non-adaptive algorithms. A non-adaptive algorithm draws m_i samples from the ith index for every $1 \le i \le n$. The sample complexity is $q = \sum_{i=1}^{n} m_i$.

▶ **Lemma 53.** *For $\delta < 1/3$, consider an algorithm that for every $i \ge 1$ draws m_i samples from the ith index, and let $q = \sum_{i=1}^{n} m_i$. The KL-divergence of runs over inputs drawn from $D_n(\delta, 0)$ and from runs over inputs drawn from $D_n(\delta, r)$ is bounded by $5r^2\delta^2 q$.*

A.3 Lower bound for adaptive algorithms

Adaptive algorithms are allowed to choose their next step based on past samples. Therefore, their analysis is more difficult. By Yao's principle [18], a probabilistic algorithm is equivalent to a distribution over deterministic algorithms. Moreover, the success probability of a probabilistic algorithm is the expected success probability over the choice of a deterministic algorithm.

Every deterministic adaptive algorithm can be described as a decision tree, where edges correspond to samples (0 and 1) and every non-leaf node is associated with the index $1 \le i \le n$ to sample in the following step. A run of a deterministic algorithm is its execution path, which corresponds to the leaf the algorithm reaches in its last step.

To show a lower bound of q samples, we have to show that for every decision tree of height q, the total-variation distance between the distribution of leaves reached when the input is drawn from $D_n(\delta, 0)$ and their distribution when the input is drawn from $D_n(\delta, r)$ is bounded by some $d < 1/3$. This implies that the success probability, which cannot exceed the maximal success probability of a deterministic algorithm, is at most $1 - d$. By Pinsker's inequality, it suffices to bound the KL-divergence of the leaf distributions by $2d^2$.

We repeatedly apply the chain-rule for KL-divergence (Lemma 62) to obtain that the KL-divergence of the leaf distributions is bounded by the expected KL-divergence of every step individually.

To analyze an adaptive algorithm, whose structure can be arbitrarily detailed, we simulate it using a stronger model whose runs have shorter descriptions. Let $m = \left\lfloor \frac{d^2}{400\delta^2} \right\rfloor$ and $q \leq \frac{d^2}{1000}mn$. At the initialization phase, we run an m-height rectangle algorithm, which is a non-adaptive algorithm that makes m queries in each index $1 \leq i \leq n$, obtaining a pre-processing sampling matrix $M \in \{0,1\}^{m \times n}$ at the cost of mn samples. For every individual index $1 \leq i \leq n$, the first m queries are answered based on the pre-processing sampling matrix. Once the algorithm makes the $m+1$st sample on the ith index, the simulator reveals the exact value of p_i, and exactly simulates further queries to the ith index using the full-knowledge about p_i.

▶ **Observation 54.** *Let $d > 0$. An adaptive algorithm that draws $q \leq \frac{d^2}{1000}mn$ samples has at most $\frac{d^2}{1000}n$ indexes sampled more than m times.*

▶ **Observation 55.** *Every run of an adaptive algorithm with $q \leq \frac{d^2}{1000}mn$ samples can be described as a tuple of the pre-processing sampling matrix M, the set I of at most $\frac{d^2}{1000}n$ fully-revealed indexes and a map $f : I \to \{+,-\}$ corresponding to the exact value of $p_i \in \frac{1\pm\delta}{2}$.*

▶ **Lemma 56.** *Let $\delta < 1/3$, $r < 1/12$, $n \leq 64/r^2$ and $d > 0$. For every adaptive algorithm with $q \leq \frac{d^4}{4\cdot 10^5 \delta^2}n$ samples, the distance between runs over inputs drawn from $D_n(\delta, 0)$ and runs over inputs drawn from $D_n(\delta, r)$ is bounded by d.*

B Common distributions and fact sheet

B.1 Common distributions

▶ **Definition 57** (Bernoulli distribution). *Bernoulli distribution* $\mathrm{Ber}(p)$ *is the distribution over* $\{0,1\}$ *for which* $\Pr[1] = p$ *and* $\Pr[0] = 1-p$.

▶ **Definition 58** (Binomial distribution). *The binomial distribution* $\mathrm{Bin}(n,p)$ *is the sum of* n *independent samples from the Bernoulli distribution* $\mathrm{Ber}(p)$.

B.2 Fact sheet

In the following we provide some well-known bounds and identities.

▶ **Lemma 59** (Pinsker's inequality). $D_{\mathrm{KL}}(\mu \parallel \tau) \geq 2(d_{\mathrm{TV}}(\mu,\tau))^2$.

▶ **Lemma 60.** *For $x \geq -2/3$, $-\ln(1+x) \leq -x + x^2$.*

▶ **Lemma 61** (see [8]). *For $p, q \in [0,1]$, $D_{\mathrm{KL}}(p,q) \leq \frac{(p-q)^2}{q(1-q)}$.*

▶ **Lemma 62** (Chain rule for D_{KL}). *Let μ and τ be two distributions over $A \times B$. In this setting, $D_{\mathrm{KL}}(\mu \parallel \tau) = D_{\mathrm{KL}}(\mu|_A \parallel \tau|_A) + \mathrm{E}_{a\sim\mu|_A}\left[D_{\mathrm{KL}}\left(\mu|^{\{a\}\times B} \,\middle\|\, \tau|^{\{a\}\times B}\right)\right]$.*

When Local and Non-Local Meet: Quadratic Improvement for Edge Estimation with Independent Set Queries

Tomer Adar
Technion – Israel Institute of Technology, Haifa, Israel

Yahel Hotam
University of Haifa, Israel

Amit Levi
University of Haifa, Israel

Abstract

We study the problem of estimating the number of edges in an unknown graph. We consider a hybrid model in which an algorithm may issue independent set, degree, and neighbor queries. We show that this model admits strictly more efficient edge estimation than either access type alone. Specifically, we give a randomized algorithm that outputs a $(1 \pm \varepsilon)$-approximation of the number of edges using $O\left(\min\left(\sqrt{m}, \sqrt{\frac{n}{\sqrt{m}}}\right) \cdot \frac{\log n}{\varepsilon^{5/2}}\right)$ queries, and prove a nearly matching lower bound. In contrast, prior work shows that in the local query model (Goldreich and Ron, *Random Structures & Algorithms* 2008) and in the independent set query model (Beame *et al.* ITCS 2018, Chen *et al.* SODA 2020), edge estimation requires $\widetilde{\Theta}(n/\sqrt{m})$ queries in the same parameter regimes. Our results therefore yield a quadratic improvement in the hybrid model, and no asymptotically better improvement is possible.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases Edge count estimation, Degree oracle, Neighbor oracle, Independent-set oracle

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.33

Category RANDOM

Related Version *Full Version*: https://arxiv.org/pdf/2601.21457

1 Introduction

Estimating the number of edges in an unknown graph is a central problem in sublinear algorithms. Formally, given query access to a graph $G = (V, E)$ on n vertices, the goal is to approximate $m = |E|$ within a $(1 \pm \varepsilon)$-multiplicative factor while making as few queries as possible. The problem captures the inherent tension between accuracy and information constraints and has been extensively studied (see e.g., [21, 22, 20, 19, 5, 23, 28, 8, 12])

A key theme in this line of work is that the achievable query complexity depends critically on the type of access allowed to the graph. Different query models expose fundamentally different kinds of information, leading to qualitatively different algorithmic techniques.

Until recently, most work in the sublinear algorithms community focused on *local access models*, where queries are restricted to a vertex and its neighborhood. Typical examples include degree queries, which return a vertex's degree, and neighbor queries, which return a specified neighbor. Such queries provide fine-grained, precise information about the graph.

Beame, Har-Peled, Ramamoorthy, Rashtchian, and Sinha [6] introduced a markedly different access model, the *independent-set (IS) oracle*. In this model, an algorithm may query any subset $S \subseteq V$, and the oracle responds with a single bit indicating whether S induces an edge-free subgraph. Unlike local queries, IS queries do not reveal the identity or


© Tomer Adar, Yahel Hotam, and Amit Levi;
licensed under Creative Commons License CC-BY 4.0

location of edges. Instead, they provide a coarse *non-local* information about the presence or absence of edges within a queried set. This model can be viewed as a form of combinatorial group testing, where a single query can simultaneously probe a large portion of the graph. As such, IS queries expose structural information that is fundamentally inaccessible through purely local inspection.

[6] first analyzed edge estimation in the IS model and obtained an upper bound of $\widetilde{O}(n^{2/3})$ queries. Chen, Levi, and Waingarten [15] refined this analysis and established nearly optimal bounds: $(1 \pm \varepsilon)$-approximate edge estimation can be performed using $\widetilde{O}\left(\min\left(\sqrt{m}, \frac{n}{\sqrt{m}}\right)\right)$ IS queries, and show that this dependence on n and m is optimal up to polylogarithmic factors.

Local queries and IS queries exhibit inherently different strengths. Local queries provide detailed information about individual vertices and their neighborhoods, whereas IS queries give coarse binary feedback over subsets. When considering the local model, (nearly) tight bounds of $\widetilde{\Theta}(n/\sqrt{m})$ queries for edge estimation are known [22, 21]. Therefore, there exist settings in which both oracles require the same asymptotic number of queries, but for entirely different reasons: local queries are limited by the need to sample enough edges from sparse regions, while IS queries are limited by the coarse granularity of global binary feedback. This illustrates that even when the query complexities coincide, the information revealed by each oracle is *fundamentally different*, reflecting their complementary powers.

This contrast naturally motivates the study of *hybrid* access models, in which an algorithm is allowed to use both local queries and IS queries. Such a model offers the potential to combine non-local aggregation with local precision: local queries can be used to identify informative regions of the graph or reveal degrees, whereas IS queries can efficiently summarize the presence of edges across large subsets. Understanding the power and limitations of this hybrid model is essential for clarifying how different forms of partial information interact, and for determining whether combining query types can lead to provable improvements over either model alone.

1.1 Our Results

In this work, we provide a tight characterization of this hybrid model.

▶ **Theorem 1** (Upper Bound). *Consider the query model consisting of IS, degree, and neighbor queries to an unknown graph $G = (V, E)$ with $|V| = n$ and $|E| = m$. For every $0 < \varepsilon < 1$, there exists a randomized algorithm that uses $O\left(\min\left(\sqrt{m}, \sqrt{\frac{n}{\sqrt{m}}}\right) \cdot \frac{\log n}{\varepsilon^{5/2}}\right)$ queries in expectation and outputs a $(1 \pm \varepsilon)$-approximation of m with probability at least $2/3$.*

▶ **Theorem 2** (Lower Bound). *For every $\varepsilon \in (0, 1)$, in the query model allowing independent set, degree, and neighbor queries, any randomized algorithm that estimates the number of edges m of an n-vertex graph to within a $(1 \pm \varepsilon)$ factor with success probability at least $2/3$ requires $\Omega\left(\min\left(\sqrt{m}, \sqrt{\frac{n}{\varepsilon\sqrt{m}}}\right)\right)$ queries.*

In particular, in regimes where each individual model would require $\widetilde{\Theta}(n/\sqrt{m})$ queries, the combination yields a *quadratic improvement*, whereas in other regimes it matches the best of the two models. These results demonstrate that IS and local queries are synergistic yet complementary, and that the hybrid model achieves the maximum possible improvement for the task of edge estimation.

1.2 Technical overview

1.2.1 Upper-Bound

Our algorithm is advice-based: it receives a guess $\bar{m}$ for the number of edges m, but treats this guess as untrusted. For each guess, the algorithm must either produce a $(1\pm\varepsilon)$-approximation to m, or determine that $\bar{m}$ is not within the correct scale. The goal is to obtain query complexity $\widetilde{O}\left(\min\left(\sqrt{m}, \sqrt{n/\sqrt{m}}\right)\right)$.

To achieve this goal, we interleave two exponentially progressing sequences of guesses. The upper sequence starts from n^2 and decreases geometrically, taking $\bar{m} = 2^{-\ell}n^2$ at iteration ℓ. The lower sequence increases geometrically, taking $\bar{m} = 2^{\ell/2}$. The different rates are chosen so that the cost of handling a guess from either sequence is comparable within the same iteration.

For a fixed guess $\bar{m}$, the algorithm partitions vertices according to a degree threshold k, classifying vertices as low-degree or high-degree. This induces a decomposition of the edge set into edges between two low-degree vertices (m_{LL}), edges between low- and high-degree vertices (m_{LH}), and edges between two high-degree vertices (m_{HH}). These components are handled separately, exploiting the fact that high-degree vertices are rare once k is chosen appropriately.

The main component is estimating m_{LL}. We adapt the edge-sampling approach of [6], which samples an induced subgraph by including each vertex independently with probability $1/\sqrt{\bar{m}}$. If all vertices were low-degree, then this estimator would have low variance. We use the degree oracle to identify edges incident to high-degree vertices and exclude them from counting to keep the variance low while preserving unbiasedness.

Edges between high-degree vertices have small contribution overall. For an appropriate choice of k, the number of such edges is $O(\varepsilon m)$, which can be absorbed into the allowed additive error. For low-high edges, we further refine the low-degree vertices by introducing a second threshold k', defining very-low-degree vertices, and only count the edges between them to high-degree vertices. With this refinement, for an appropriate choice of k, the number of excluded edges is $O(\varepsilon m)$, allowing us to only consider edges between very-low-degree vertices and high-degree vertices.

The remaining quantity, denoted m_{L_1H} (between very-low-degree vertices and high-degree vertices), is estimated via a weighted sampling procedure. We sample a vertex, then sample one of its neighbors, and count the edge with weight proportional to the degree of the sampled vertex whenever it is very-low-degree and its neighbor is high-degree. This yields an unbiased estimator whose variance is controlled using $O(n/(\varepsilon^3 k))$ samples.

We analyze the algorithm separately for the two guess sequences. We first consider the upper guess sequence, for which $\bar{m} \geq m$ holds throughout until a correct-scale guess is reached. In this regime, we set $k = \Theta\left(\sqrt{n\sqrt{\bar{m}}/\varepsilon}\right)$, except in the very dense case $\bar{m} = \Omega(\varepsilon n^2)$, where a separate optimization applies. With this choice, the combined cost of estimating m_{LL} and m_{L_1H} is $O\left(\frac{\log n}{\varepsilon^{5/2}} \cdot \sqrt{\frac{n}{\sqrt{\bar{m}}}}\right)$.

When $\bar{m} = \Theta(m)$, the algorithm outputs $\tilde{m} = (1 \pm O(\varepsilon))m$ with high probability. More generally, when $\bar{m}$ is larger than the true value, the output is noticeably smaller than $\bar{m}$, and in particular bounded by $m + \varepsilon\bar{m} \ll \bar{m}$ when $\bar{m} \gg m$. This gap allows the algorithm to reliably detect overly large guesses and continue to smaller ones.

For the lower guess sequence, the algorithm cannot assume that $\bar{m} \geq m$. Instead, it sets the degree threshold to $k = \min(\bar{m}, n-1)$ and attempts to verify that no vertex exceeds this degree. If this verification succeeds, then all vertices are low-degree and $m_{LL} = m$, so

estimating m_{LL} alone suffices. If the guess $\bar{m}$ is too small, then either $m_{LL} > 2\bar{m}$, or there exist $\Omega(m)$ edges adjacent to vertices of degree larger than $\bar{m}$ (in particular, there exists a vertex of degree larger than $\bar{m}$). These cases are detected by running a complexity-safe version of the m_{LL} estimator, which aborts if it exceeds the cost expected when assuming that $\bar{m} \geq m$, and by searching for a high-degree vertex. The resulting cost per lower-sequence iteration is $O\left(\frac{\log n}{\varepsilon^2}\sqrt{\bar{m}}\right)$.

Standard amplification could be applied to each iteration to ensure correctness with high probability. However, a more refined analysis shows that such amplification is unnecessary. The failure probability of an upper-sequence iteration is $O(m/\bar{m})$, while that of a lower-sequence iteration is $O(\bar{m}/m)$. Since both guess sequences are exponential, the total failure probability accumulated over all iterations preceding the one with $\bar{m} \approx m$ remains bounded by a constant. The iteration for which $\bar{m} = \Theta(m)$ therefore succeeds with constant probability.

Finally, while the above discussion yields the desired query complexity with high probability, controlling the expected complexity requires care due to the possibility that "the" correct guess fails, and then further high-sequence guesses are smaller than m and low-sequence guesses are greater than m, making the rest of the run incompatible with the correctness analysis (even though complexity analysis still holds). To address this, we execute the guesses using an iterative deepening schedule. We first consider iteration 0, then iterations $(0, 1)$, then $(0, 1, 2)$, and so on. This scheduling ensures that the tail of the running-time distribution decays sufficiently fast, so that the expected query complexity matches the high-probability bound asymptotically.

1.2.2 Lower bound

We show that any deterministic algorithm with access to degree, neighbor and independent-set oracles cannot distinguish between two distributions over graph whose degrees are separated by a $(1+\Omega(\varepsilon))$-factor, unless it makes $\Omega\left(\min\left(\sqrt{m}, \sqrt{n/\varepsilon\sqrt{m}}\right)\right)$ queries in expectation (note that the input distribution is the source of randomness). By Yao's principle [29], this lower bound also applies to probabilistic algorithms with the same expected query complexity.

Canonical graph construction

To define the canonical graph $G_{n,d,\varepsilon}$, let $d_0(n, \epsilon) \stackrel{\text{def}}{=} 1/(n^{1/3}\epsilon^{2/3})$. For the simplicity of the overview, we focus on the case where $d \geq d_0(n, \varepsilon)$, whose lower bound is $\Omega(\sqrt{n/\varepsilon\sqrt{m}})$. The input graph consists of disjoint vertex sets K, L, H, and A whose sizes are:

$$n_k = \left\lfloor \sqrt{2nd} \right\rfloor, \quad n_h = \left\lceil \varepsilon^{1/2} n^{1/4} d^{3/4} \right\rceil, n_\ell = \left\lceil \varepsilon^{1/2} n^{3/4} d^{1/4} \right\rceil, \quad n_a = n - (n_k + n_h + n_\ell).$$

For the graph edges: the set K forms a clique, the sets L and H are the sides of a biclique and A consists of isolated vertices. The K-clique contributes $\Omega(m)$ edges (but not more than m), and the L-H graph contributes $\Theta(\varepsilon m)$ edges. Therefore, the number of edges is $\Theta(m)$ and the average degree is $\Theta(d)$.

Graph distributions

In the first distribution we permute the vertices of $G_{n,d,0}$, that is, the graph only consists of a clique but not a biclique, and it has at most m edges. In the second distribution we permute the vertices of $G_{n,d,\varepsilon}$, which has at least εm additional edges.

Intuition

As long as the algorithm does not query a vertex in the biclique or find a biclique-edge, the input distributions are indistinguishable. For local queries, the probability of hitting the biclique scales with the size of L. For independent-set queries, a query must include at least one vertex from H to detect an edge in the biclique. Large IS-queries are limited by the clique size $|K|$, since including two vertices from K makes the queried set not-independent regardless of whether or not it has a biclique-edge.

A more careful argument shows that for degree and neighbor queries, the probability of sampling a biclique vertex is $O((n_\ell + n_h)/n) = O(n_\ell/n)$. Hence, $\Omega\left(\frac{n}{n_\ell}\right) = \Omega\left(\sqrt{\frac{n}{\varepsilon\sqrt{m}}}\right)$ queries are needed to hit a biclique vertex with $\Omega(1)$ probability.

For IS-queries, the clique K limits the effective query size to $O(n/n_k) = O(\sqrt{n/d})$, and the probability of hitting H using such query is $O\left(\frac{n_h}{n} \cdot \sqrt{\frac{n}{d}}\right) = O\left(\frac{n_h}{\sqrt{nd}}\right)$. This probability leads to the same lower bound as before. Overall, as $d = \Theta(m/n)$, distinguishing the two distributions with probability $\Omega(1)$ requires $\Omega\left(\sqrt{\frac{n}{\varepsilon\sqrt{m}}}\right)$ queries, which establishes the desired lower bound.

For $d < d_0(n, \epsilon)$, we observe that the definition of n_h is fixed to 1, and therefore, we can reduce n_ℓ to $\lceil \varepsilon nd \rceil$. Besides the alternative value of n_ℓ, the construction is the same. Analyzing this small-d case uses similar ideas and results in an $\Omega(\sqrt{m})$ lower bound.

1.3 Related work

A generalization of independent set queries appears in a broader line of work investigating the relationship between decision and counting problems [25, 26, 24, 16, 17, 11]. Independent set queries also have deep connections to group testing, where one tests large collections of items to detect the presence of rare events [18, 27, 14]. Independent set queries also have close connections to learning theory. In particular, a classical line of work on learning hidden graphs studies *edge-detecting queries*, which are equivalent to independent set queries [3, 4, 1].

Several works have explored combinatorial or global query primitives beyond purely local access. Beyond standard IS queries, Beame *et al.* [6] introduced *bipartite independent set (BIS) queries*, which ask whether there exists at least one edge between two given vertex subsets, and used them to obtain tight sublinear bounds for edge count estimation. Subsequently, [2] designed *non-adaptive* algorithms for edge counting and sampling in the BIS model. The BIS model was later extended to more general settings (see e.g., [9, 13, 10]).

Perhaps the work most closely related to ours is that of Beretta, Chakrabarty, and Seshadhri [7], which studies average-degree estimation under a query model combining random-edge sampling with structural queries (e.g., pair queries and full neighborhood access). They also discuss a structural oracle that, given a set $S \subseteq V$, returns the number of edges induced by S at a cost proportional to $|S|$. These structural queries provide precise local or aggregate information but follow a different cost model than IS queries. In particular, it is possible to show that IS oracle can simulate their structural oracle using (almost) linear number of IS queries.

2 Preliminaries

Given a positive integer n, we denote $[n] = \{1, ..., n\}$. For a set A, we denote the power set of A by 2^A. All graphs $G = (V, E)$ considered in this paper are undirected and simple (meaning that there are no parallel edges or loops), with $V = [n]$ as their vertex set (we

often just write $G = ([n], E)$). We also let $m = |E|$ be the number of edges in the graph and $d = 2m/n$ be the average degree in the graph. We use $\deg(v)$ to denote the degree of a vertex $v \in [n]$. We use boldface letters (such as $\mathbf{X}$) to denote random variables. We use the notation $\tilde{x} = (1 \pm \varepsilon)x$ to denote that $(1-\varepsilon)x \leq \tilde{x} \leq (1+\varepsilon)x$.

An algorithm is allowed to perform three types of queries to access the (otherwise hidden) edge set of the graph E, as defined by the following oracles:

▶ **Definition 3** (Degree oracle). *Given an undirected graph $G = ([n], E)$, its* degree oracle *is a map $\mathsf{Deg} : [n] \to [n-1] \cup \{0\}$, which for a vertex $u \in [n]$ returns the degree* $\deg(u)$.

▶ **Definition 4** (Random Neighbor oracle). *Given an undirected graph $G = ([n], E)$, its* random neighbor oracle *is a map $\mathsf{Neigh} : [n] \to [n]$, which for a vertex $u \in [n]$ returns a uniformly random neighbor of u. If u has no neighbors, it returns a special character to signify it.*

▶ **Definition 5** (Independent set oracle). *Given an undirected graph $G = ([n], E)$, its* independent set oracle *is a boolean map $\mathsf{IS} : 2^{[n]} \to \{0, 1\}$, which for a subset of vertices $U \subseteq [n]$ returns 1 if and only if U is an independent set of G (i.e., the subgraph induced on U contains no edges).*

Some of our algorithms will include the assumption that $m \geq 1$, note that we can distinguish this from the case that $m = 0$ using a single query to IS.

2.1 Classifying vertices and edges

We classify the vertices in V into three disjoint sets by their degrees, and use those sets to classify the edges in the graph:

▶ **Definition 6.** *Given a graph $G = (V, E)$, let k' and k be two degree thresholds (not necessarily integers) for which $k \geq k' \geq 0$ (where $n = |V|$). Let V_{L_1}, V_{L_2}, V_H be a partition of $[n]$ into disjoint sets such that:*

$$V_{L_1} = \{u \in V : \deg(u) \leq k'\}, V_{L_2} = \{u \in V : k' < \deg(u) \leq k\}, V_H = \{u \in V : \deg(u) > k\}.$$

Also, let $V_L = V_{L_1} \cup V_{L_2} = \{u \in V : \deg(u) \leq k\}$.

For two labels $\sigma_1, \sigma_2 \in \{L_1, L_2, L, H\}$, we use $E_{\sigma_1\sigma_2}$ to denote the set of edges between V_{σ_1} and V_{σ_2} (note that $E_{\sigma_1\sigma_2}$ and $E_{\sigma_2\sigma_1}$ are identical). Also, we let $m_{\sigma_1\sigma_2} = |E_{\sigma_1\sigma_2}|$ be the number of these edges.

At top-level, we "guess" (formally, search for) an advice $\bar{m}$, hopefully just slightly greater than the correct answer m. Based on $\bar{m}$ we define the threshold degrees k and k', and use different logic to estimate the number of edges between every pair of subsets. For some pairs the estimation is zero, since the choice of k, k' means that the number of relevant edges is no more than linear in the allowed additive error and thus are negligible.

For $\bar{m} > 0$ and $0 < \varepsilon < 1$, let $k(\bar{m}, \varepsilon) \stackrel{\text{def}}{=} \sqrt{\frac{2n\sqrt{\bar{m}}}{\varepsilon}}$ and $k' = \min(k, \bar{m}/\varepsilon k)$, and recall the sets of Definition 6. Clearly, both k and $\bar{m}/\varepsilon k$ are monotone increasing in $\bar{m}$ (when n and ε are considered fixed). Therefore, the set $V_H(\bar{m}, \varepsilon)$ and the union set $V_{L_2}(\bar{m}, \varepsilon) \cup V_H(\bar{m}, \varepsilon)$ are monotone non-increasing in $\bar{m}$. Note that when $k' = k$, we have $|V_{L_2}| = 0$ and thus $V_L = V_{L_1}$.

▶ **Lemma 7.** *Let $k(\bar{m}, \varepsilon) = \sqrt{2n\sqrt{\bar{m}}/\varepsilon}$. If $\bar{m} \geq \frac{1}{4}m$, then the number of edges between $V_{L_2} \cup V_H$ and V_H is at most $64\varepsilon\bar{m}$.*

▶ **Lemma 8.** *If $\bar{m} \geq \frac{1}{4}m$ and $k(\bar{m}, \varepsilon) = \sqrt{2n\sqrt{\bar{m}}/\varepsilon}$, then $m_{LL} + m_{L_1H} \geq m - 64\varepsilon\bar{m}$.*

3 Upper Bound

In this section, we prove an upper bound on the number of queries an algorithm with access to Deg, Neigh, IS needs to make in order to achieve a $1 \pm \varepsilon$ estimation of the number of edges in a graph. Specifically, we prove the following:

▶ **Theorem 9.** *There exists an algorithm Estimate-Edges(n, ε, G) that takes as input integer $n \geq 1$, parameter $\varepsilon \in (0, 1]$, and access to the oracles Deg, Neigh, IS of a graph $G = ([n], E)$ with $m = |E|$. Estimate-Edges makes an expected number of $O\left(\frac{\log n}{\varepsilon^{5/2}} \min\left(\sqrt{\frac{n}{\sqrt{m}}}, \sqrt{m}\right)\right)$ queries to the oracles and outputs a number $\tilde{\boldsymbol{m}}$ such that with probability at least $2/3$ satisfies $(1-\varepsilon)m \leq \tilde{\boldsymbol{m}} \leq (1+\varepsilon)m$.*

We prove Theorem 9 through Lemma 22, which is logically equivalent. We use the parameters and notation described in Table 1 throughout the paper.

Table 1 Notation and parameters used throughout this section.

Symbol	Description
$G = (V, E)$	The input graph.
n	Number of vertices in the graph ($n = \|V\|$).
m	Number of edges in the graph ($m = \|E\|$).
$\bar{m}$	Untrusted advice for the number of edges m, where $\bar{m} > 0$.
$\tilde{\boldsymbol{m}}$	Random variable output of an estimation procedure.
k	Degree threshold: vertices of degree at most k are *low-degree*, and vertices of degree greater than k are *high-degree*, with $k \geq 0$.
k'	Secondary degree threshold, satisfying $0 \leq k' \leq k$.

3.1 Estimating m_{LL} (with advice)

We restate the following lemma from [6]:

▶ **Lemma 10** ([6])**.** *There exists a deterministic procedure Enumerate-Edges whose input is a graph $G = (V, E)$ accessible through the independent-set oracle and a vertex set $A \subseteq V$, and its output is a list of all edges in the subgraph induced on A's vertices. The query complexity of this procedure is $O(1 + |E_A| \log n)$, where $n = |V|$ is the number of vertices and E_A is the set of edges in the subgraph induced on A's vertices.*

Procedure Sample-LL-Edges-Sparse, implemented in Algorithm 1, is an unbiased estimator for $\frac{m_{LL}}{\bar{m}}$. It draws a set $\mathbf{S} \subseteq V$ that every vertex belongs to with probability $1/\sqrt{\bar{m}}$, and then enumerates the edges in the subgraph induced on $\mathbf{S}$. Only edges between two low-degree vertices $(\deg(u), \deg(v) \leq k)$ are counted.

▶ **Lemma 11.** *The output of Procedure Sample-LL-Edges-Sparse (Algorithm 1) is a random variable whose expected value is $\frac{m_{LL}}{\bar{m}}$ and variance is at most $\left(1 + \frac{2k}{\sqrt{\bar{m}}}\right)\frac{m_{LL}}{\bar{m}}$. Moreover, the procedure's expected query complexity is $O(1 + \frac{m}{\bar{m}} \log n)$.*

In procedure Estimate-LL-Edges-Advice (Algorithm 2), we estimate m_{LL} by averaging the outcomes of $O\left(\frac{1+k/\sqrt{\bar{m}}}{\varepsilon^2}\right)$ invocations of Sample-LL-Edges-Sparse. Since each invocation has expected value $m_{LL}/\bar{m}$, this averaging yields good concentration around the mean.

Algorithm 1: Procedure Sample-LL-Edges-Sparse$(G, \bar{m}, k)$.

Input: Integer $n \geq 1$, parameters $k \geq 0$, $\bar{m} > 0$.
Input: Access to the oracles IS, Deg of a graph $G = ([n], E)$ with $m = |E|$.
Output: An unbiased estimation $\mathbf{X}$, for $\frac{m_{LL}}{\bar{m}}$.

1. Draw a subset $\mathbf{S} \subseteq V$ that every vertex belongs to with probability $1/\sqrt{\bar{m}}$, independently.
2. Run Enumerate-Edges$(G, \mathbf{S})$, and let $E_{\mathbf{S}}$ be the resulting list of edges.
3. Initialize $\mathbf{X} \leftarrow 0$.
4. For each edge $(\boldsymbol{u}, \boldsymbol{v}) \in E_{\mathbf{S}}$:
   a. Let $d_{\boldsymbol{u}} \leftarrow \deg(\boldsymbol{u})$ (via Deg).
   b. Let $d_{\boldsymbol{v}} \leftarrow \deg(\boldsymbol{v})$ (via Deg).
   c. If $d_{\boldsymbol{u}} \leq k$ and $d_{\boldsymbol{v}} \leq k$:
      i. Increment $\mathbf{X} \leftarrow \mathbf{X} + 1$.
5. Return $\mathbf{X}$.

Algorithm 2: Procedure Estimate-LL-Edges-Advice$(n, \varepsilon, G, \bar{m}, k)$.

Input: Integer $n \geq 1$, parameters $\varepsilon \in (0, 1], k \geq 0$, $\bar{m} > 0$.
Input: Access to the oracles IS, Deg of a graph $G = ([n], E)$ with $m = |E|$.
Output: A number $\tilde{\boldsymbol{m}}_{LL}$.

1. Let $\boldsymbol{c} \leftarrow 0$, $q \leftarrow \left\lceil \frac{600}{\varepsilon^2} \max\left(1, \frac{k}{\sqrt{\bar{m}}}\right) \right\rceil$.
2. For $i = 1, ..., q$:
   a. Run Sample-LL-Edges-Sparse$(n, G, \bar{m}, k)$ and let $\mathbf{X}_i$ be its result.
   b. Increment $\boldsymbol{c} \leftarrow \boldsymbol{c} + \frac{1}{q}\mathbf{X}_i$.
3. Return $\tilde{\boldsymbol{m}}_{LL} = \bar{m} \cdot \boldsymbol{c}$.

▶ **Lemma 12.** *Let $\tilde{\boldsymbol{m}}_{LL}$ be the (random) output of procedure Estimate-LL-Edges-Advice (Algorithm 2).*

- *If $\bar{m} \geq \frac{1}{4}m_{LL}$, then with probability at least $1 - \frac{m_{LL}}{200\bar{m}}$, $|\tilde{\boldsymbol{m}}_{LL} - m_{LL}| \leq \varepsilon\bar{m}$.*
- *If $\bar{m} < m_{LL}$, then with probability at least $1 - \frac{\bar{m}}{200m_{LL}}$, $\tilde{\boldsymbol{m}}_{LL} > (1 - \varepsilon)\bar{m}$.*

Moreover, the expected query complexity is $O\left(\frac{\log n}{\varepsilon^2}\left(1 + \frac{k}{\sqrt{\bar{m}}}\right)\left(1 + \frac{m}{\bar{m}}\right)\right)$. Note that if $\frac{1}{4}m_{LL} \leq \bar{m} < m_{LL}$, then both guarantees hold simultaneously.

3.2 Estimating m_{LH} (with advice)

In this subsection we provide an estimation algorithm for the number of edges between low-degree vertices ($\deg(u) \leq k$) and high-degree vertices ($\deg(u) > k$). Recall that the number of low-high edges is denoted by m_{LH}, and the set of these edges is denoted by E_{LH}.

We define the "very-low" degree threshold as $k' = \min(k, \bar{m}/\varepsilon k)$. Let V_{L_1} denote the set of vertices of degree at most k', let E_{L_1H} be the set of edges between vertices in V_{L_1} and high-degree vertices, and let $m_{L_1H} = |E_{L_1H}|$. Since $V_{L_1} \subseteq V_L$, it follows that $m_{L_1H} \leq m_{LH}$. Moreover, Lemma 7 implies that the number of edges between high-degree vertices and vertices of degree exceeding k' is at most $4\varepsilon\bar{m}$. Consequently, $m_{L_1H} \leq m_{LH} \leq m_{L_1H} + 4\varepsilon\bar{m}$.

The estimator for m_{L_1H}, implemented as procedure Estimate-L_1H-Edges-Advice (Algorithm 3), averages over $O(nk'/\varepsilon^2\bar{m})$ samples as follows: we uniformly draw a vertex $\boldsymbol{u}$ and a neighbor $\boldsymbol{v}$ of $\boldsymbol{u}$ (if $\deg(\boldsymbol{u}) = 0$, we proceed to the next sample). If $\boldsymbol{u}$'s degree is "very-low" (i.e., $\deg(\boldsymbol{u}) \le k'$) and $\boldsymbol{v}$'s degree is high ($\deg(\boldsymbol{v}) > k$), then we add $\deg(\boldsymbol{u})$ to a counter $\boldsymbol{c}$. For all other cases, we do not increment the counter. The result is normalized by an n factor.

Algorithm 3: Procedure Estimate-L_1H-Edges-Advice$(n, \varepsilon, G, \bar{m}, k)$.

Input: Integers $n \ge 1$, parameters $\varepsilon \in (0, 1], k \ge 0$, $\bar{m} > 0$.
Input: Access to the oracles Deg, Neigh of a graph $G = ([n], E)$ with $m = |E|$.
Output: An unbiased estimator $\tilde{\boldsymbol{m}}_{L_1H}$ for m_{L_1H}.

1. Initialize $\boldsymbol{c} \leftarrow 0$.
2. Let $k' \leftarrow \min\left(k, \frac{\bar{m}}{\varepsilon k}\right)$.
3. Let $t \leftarrow \left\lceil \frac{200nk'}{\varepsilon^2\bar{m}} \right\rceil$.
4. For $i = 1, ..., t$:
   a. Sample a vertex $\boldsymbol{u} \in V$ uniformly at random.
   b. Sample a neighbor $\boldsymbol{v}$ of $\boldsymbol{u}$ uniformly at random (via Neigh).
   c. Let $\boldsymbol{d}_u \leftarrow \deg(\boldsymbol{u})$ (via Deg).
   d. Let $\boldsymbol{d}_v \leftarrow \deg(\boldsymbol{v})$ (via Deg).
   e. If $\boldsymbol{d}_u \le k'$ and $\boldsymbol{d}_v > k$:
      i. Increment $\boldsymbol{c} \leftarrow \boldsymbol{c} + \boldsymbol{d}_u$.
5. Return $\tilde{\boldsymbol{m}}_{L_1H} = \frac{n}{t}\boldsymbol{c}$.

▶ **Lemma 13.** *Let $\tilde{\boldsymbol{m}}_{L_1,H}$ be the (random) output of Estimate-L_1H-Edges-Advice (Algorithm 3) when executed with parameters $(G, \varepsilon, \bar{m}, k)$.*

- *If $\bar{m} \ge \frac{1}{4}m_{L_1H}$, then with probability at least $1 - \frac{m_{L_1H}}{200\bar{m}}$, $|\tilde{\boldsymbol{m}}_{L_1H} - m_{L_1H}| \le \varepsilon\bar{m}$.*
- *If $\bar{m} < m_{L_1H}$, then with probability at least $1 - \frac{\bar{m}}{200m_{L_1H}}$, $\tilde{\boldsymbol{m}}_{L_1H} > (1-\varepsilon)\bar{m}$.*

Moreover, the query complexity is bounded by $O(n/\varepsilon^3 k)$.

3.3 Handling the case where $\bar{m} < m$

Beame's algorithm (Enumerate-Edges) is meant to be executed as a whole. For our algorithm, we need a more fine-grained analysis.

▶ **Lemma 14.** *Enumerate-Edges (Lemma 10) can be implemented with an additional guarantee: for every $1 \le t \le |E_A|$, the t-th edge is added to the list after performing at most $O(1+t\log n)$ queries. In other words, it has $O(1)$ initialization phase and amortized $O(\log n)$ "next-element" phase.*

Procedure Small-$\bar{m}$-Guard (Algorithm 6) separates the regime in which the advice $\bar{m}$ is valid ($\bar{m} \ge m$) from the regime in which it is substantially underestimated ($\bar{m} < m/4$) or when it is structurally wrong ($\max_u \deg(u) > \bar{m}$). In the former case, the procedure accepts with probability at least $1 - 1/1000$ (completeness), while in the latter it rejects with probability at least $1 - \bar{m}/1000m$ (soundness). The procedure searches for vertices whose degree exceeds $\bar{m}$ and applies a moderated variant of Sample-LL-Edges-Sparse in order to reject instances in which the number of edges is significantly larger than what is consistent with the assumption $\bar{m} \ge m$.

For readability, we provide a parameterized algorithm and a corresponding parameterized analysis for completeness $1 - c$ and soundness $1 - c\bar{m}/m$, where $c \leq 1$. For the lemma statement, we use $c = 1/1000$.

Case 0 (trivial)

$m = O(\text{poly}(1/c))$. This is an easy case since we can just look for $O(\text{poly}(1/c))$ edges at the cost of $O_c(\log n)$ queries to find out whether or not $\bar{m} \geq \min(m, \text{poly}(1/c)) = m$.

Case I (typical)

$m_{LL} \geq \frac{3}{4}m$, where the threshold degree k is defined as $\min(\bar{m}, n-1)$.

Case II (very-high degree)

There exists a vertex of degree at least $\sqrt{\bar{m}}\ln(m/c)$.

Case III (sparse)

$m = \Omega(\text{poly}(1/c))$ and $m_{LL} < \frac{3}{4}m$ (where the threshold degree k is defined as $\min(\bar{m}, n-1)$) and the maximum degree is at most $\sqrt{\bar{m}}\ln(m/c)$.

Observe that Case I is "quantitative" (to be resolved by large-deviation inequalities), whereas Case II is a "qualitative" (to be resolved by looking for a witness), and therefore, they are treated using different algorithms. Case III is analyzed through the qualitative algorithm resolving Case II.

3.3.1 Quantity test (Case I)

In the quantity test (Guard-Quantity, Algorithm 4) we execute an edge sampler similar to Estimate-LL-Edges-Advice (Algorithm 2), but we use the lazy variant of Enumerate-Edges (Lemma 14) to keep the query complexity low even if $\bar{m}$ is significantly smaller than m. Additionally, if we happen to encounter a vertex whose degree is greater than $\bar{m}$, then we reject immediately.

▶ **Observation 15.** *Considering Algorithm 4 (Guard-Quantity), ignoring the possibility of an early termination (but without counting the high-degree-adjacent edges that would cause it), the gain in* $\mathbf{X}$ *in every iteration distributes the same as in Algorithm 1 (Sample-LL-Edges-Sparse).*

▶ **Observation 16.** *Assume that* $\bar{m} \geq 1$. *Considering Algorithm 4 (Guard-Quantity), ignoring the possibility of an early termination (but without counting the high-degree-adjacent edges that would cause it), the gain in* $\mathbf{X}$ *in every iteration has expected value* $\frac{m_{LL}}{\bar{m}}$ *and its variance is bounded by* $3\sqrt{\bar{m}} \cdot \frac{m_{LL}}{\bar{m}}$.

3.3.2 Quality test (Case II, Case III)

In the quality test (Guard-Quality, Algorithm 5) we use a similar logic to the quantitative test, but instead of having independent iterations, we choose the sets such that every vertex belongs to exactly one iteration. This guarantees that every potential witness belongs to the union of the iteration sets, which is not guaranteed in Algorithm 4. We reject as soon as we encounter a vertex whose degree is strictly greater than $\bar{m}$, or when we find significantly more edges than what we expect to if we would run Estimate-LL-Edges-Advice.

Algorithm 4: Procedure Guard-Quantity$(c; G, \bar{m})$.

Complexity: $O(\frac{1}{c}\sqrt{\bar{m}}\log n)$.
Completeness: If $\bar{m} \geq m$, then we accept with probability at least $1 - c/2$.
Soundness: If $\bar{m} < \frac{1}{4}m$ and $m_{LL} \geq \frac{3}{4}m$, then we reject with probability at least $1 - \frac{c\bar{m}}{m}$.
Soundness: If $m_{LL} < \frac{1}{2}m$ and the maximum degree in G is less than $\sqrt{\bar{m}}\ln(m/c)$, then we reject with probability at least $1 - \frac{c\bar{m}}{m}$.

1. Initialize $\mathbf{X} \leftarrow 0$.
2. Let $t \leftarrow \lceil \frac{96}{c}\sqrt{\bar{m}} \rceil$.
3. For t times:
   a. Draw a set $\mathbf{S} \subseteq V$ that every vertex belongs to with probability $\frac{1}{\sqrt{\bar{m}}}$, iid.
   b. Initialize Enumerate-Edges, let $L_{\mathbf{S}}$ be the result, accessible sequentially.
   c. While $L_{\mathbf{S}}$ has an unfeteched edge:
      i. Fetch $(\boldsymbol{u}, \boldsymbol{v}) \in L_{\mathbf{S}}$.
      ii. Let $d_{\boldsymbol{u}} \leftarrow \deg(\boldsymbol{u})$ via Deg query.
      iii. Let $d_{\boldsymbol{v}} \leftarrow \deg(\boldsymbol{v})$ via Deg query.
      iv. If $d_{\boldsymbol{u}} > \bar{m}$ or $d_{\boldsymbol{v}} > \bar{m}$:
         A. Return REJECT.
      v. Increment $\mathbf{X} \leftarrow \mathbf{X} + 1$.
      vi. If $\mathbf{X} \geq (5/4)t$:
         A. Return REJECT.
4. Return ACCEPT.

3.3.3 The guard procedure

The guard procedure is merely a conjunction of a deterministic fixed-count edge enumeration (to detect Case 0), the quantitative guard procedure (to detect Case I) and the qualitative procedure (to detect Case II and Case III).

3.4 Estimating m (with advice)

We combine the previously described partial-estimation procedures into two estimation procedures: Estimate-Edges-Advice-High, which is optimized for large ms, and Estimate-Edges-Advice-High, which is optimized for small ms.

In Estimate-Edges-Advice-High, we use the threshold degree $k = \Theta(\sqrt{n\sqrt{\bar{m}}/\varepsilon})$, unless $m = \Omega(\varepsilon n^2)$, in which case we use $k = n - 1$.

▶ **Lemma 17.** *Let $\tilde{\boldsymbol{m}}$ be the (random) output of Estimate-Edges-Advice-High (Algorithm 7) when given parameters ε, $G = (V, E)$ and $\bar{m}$. If $\bar{m} \geq \frac{1}{4}m$, then $|\tilde{\boldsymbol{m}} - m| \leq 70\varepsilon\bar{m}$ with probability at least $1 - \frac{m}{100\bar{m}}$.*

▶ **Lemma 18.** *Let $\tilde{\boldsymbol{m}}$ be the (random) output of Estimate-Edges-Advice-Low (Algorithm 8) when given parameters ε, $G = (V, E)$ and $\bar{m}$. The following is guaranteed for $m = |E|$ (unknown to the algorithm):*

- *If $\bar{m} \geq m$, then with probability at least $1 - \frac{m}{200\bar{m}} - \frac{1}{1000}$, $|\tilde{\boldsymbol{m}} - m| \leq \varepsilon\bar{m}$.*
- *If $\bar{m} \geq \frac{1}{4}m$, then with probability at least $1 - \frac{m}{200\bar{m}}$, $|\tilde{\boldsymbol{m}} - m| \leq \varepsilon\bar{m}$ or $\tilde{\boldsymbol{m}} > (1 - \varepsilon)\bar{m}$.*
- *$1 \leq \bar{m} < m$, then with probability at least $1 - \frac{\bar{m}}{200m}$, $\tilde{\boldsymbol{m}} > (1 - \varepsilon)\bar{m}$.*

Algorithm 5: Procedure Guard-Quality$(c; G, \bar{m})$.

Complexity: $O(\frac{1}{c}\sqrt{\bar{m}}\log n)$.
Completeness: If $\bar{m} \geq m$, then we accept with probability at least $1 - c/2$.
Soundness: If $\max_u \deg(u) \geq \sqrt{\bar{m}}\ln(m/c)$, then we reject with probability at least $1 - \frac{c\bar{m}}{m}$.

1. Let $t \leftarrow \lceil\sqrt{\bar{m}}\rceil$.
2. For each $u \in V$:
   a. Draw $\boldsymbol{i}_u \in \{1, \ldots, t\}$ uniformly and independently.
3. Initialize $\mathbf{X} \leftarrow 0$.
4. For i from 1 to t:
   a. Let $\mathbf{S}_i \leftarrow \{u \in V : \boldsymbol{i}_u = i\}$.
   b. Initialize Enumerate-Edges, let $L_{\mathbf{S}}$ be the result, accessible sequentially.
   c. While $L_{\mathbf{S}}$ has an unfeteched edge:
      i. Fetch $(\boldsymbol{u}, \boldsymbol{v}) \in L_{\mathbf{S}}$.
      ii. Let $d_{\boldsymbol{u}} \leftarrow \deg(\boldsymbol{u})$ via Deg query.
      iii. Let $d_{\boldsymbol{v}} \leftarrow \deg(\boldsymbol{v})$ via Deg query.
      iv. If $d_{\boldsymbol{u}} > \bar{m}$ or $d_{\boldsymbol{v}} > \bar{m}$:
         A. Return REJECT.
      v. Increment $\mathbf{X} \leftarrow \mathbf{X} + 1$.
      vi. If $\mathbf{X} \geq 2t/c$:
         A. Return REJECT.
5. Return ACCEPT.

3.5 Estimating m without any advice

In this subsection we introduce our main procedure, Estimate-Edges. At a high level, the algorithm begins with an extreme initial estimate for $\bar{m}$ and iteratively refines it toward an accurate value of m. A straightforward serial execution of these iterations (a simple for-loop) achieves correctness with high probability, but its query complexity is small only with high probability rather than in expectation. Therefore, we execute the iterations in an iterative-deepening schedule, i.e., in rounds where we run the prefixes (0), (0, 1), (0, 1, 2), and so on.

We start by presenting a single iteration of the algorithm (Algorithm 9). We then give an iteration-bounded variant, Estimate-Edges-Bounded (Algorithm 10), which attains a correct estimate with high probability once the number of iterations is sufficiently large; however, this version guarantees low query complexity only with high probability, not in expectation.

Finally, we provide the iteration-unbounded version, Estimate-Edges (Algorithm 11), which preserves correctness while ensuring that the expected query complexity remains small.

To analyze Estimate-Edges-Iteration (and its callers) when executed on an input graph G with n vertices (where n is known to the algorithm) and m edges (unknown to the algorithm), we define the following parameters:

- $\ell_{\text{big}}(G)$ – the unique integer for which $\sqrt{2}m \leq 2^{-\ell}n^2 < \sqrt{8}m$.
- $\ell_{\text{small}}(G)$ – the unique integer for which $\sqrt{2}m \leq 2^{\ell/2} < 2m$.
- $\ell_0(G) = \min(\ell_{\text{big}}(G), \ell_{\text{small}}(G))$.

For convenience, we usually omit the G parameter and refer to ℓ_{big}, ℓ_{small} and ℓ_0.

Algorithm 6: Procedure Small-$\bar{m}$-Guard$(G, \bar{m})$.

Complexity: $O(\sqrt{\bar{m}} \log n)$.
Completeness: If $\bar{m} \geq m$ and $\max_u \deg(u) \leq \bar{m}$, then we accept with probability at least $1 - \frac{1}{1000}$.
Soundness: If $\bar{m} < \frac{1}{4}m$, then we reject with probability at least $1 - \frac{\bar{m}}{1000m}$.
Soundness: If $\max_u \deg(u) > \bar{m}$, then we reject with probability at least $1 - \frac{\bar{m}}{1000m}$.

1. Let $c \leftarrow 1/1000$.
2. Let $\bar{m}^* \leftarrow \max(16(1 + \ln c^{-1})^{14}, 1/c)$.
3. Initialize Enumerate-Edges for the whole graph, let L be the result list, accessible sequentially.
4. Initialize $X \leftarrow 0$.
5. While $X \leq \bar{m}^*$ and L has an unfeteched edge:
   a. Consume an arbitrary edge of L.
   b. Increment $X \leftarrow X + 1$.
6. If $X > \bar{m}$:
   a. Return REJECT.
7. Run Guard-Quantity$(c; G, \bar{m})$.
8. If The quantity guard rejects:
   a. Return REJECT.
9. Run Guard-Quality$(c; G, \bar{m})$.
10. If The quality guard rejects:
    a. Return REJECT.
11. Return ACCEPT.

Algorithm 7: Procedure Estimate-Edges-Advice-High$(\varepsilon, G, \bar{m})$.

Input: A graph $G = (V, E)$ (V is explicitly given) accessible through Deg, Neigh, IS.
Input: An accuracy parameter $\varepsilon > 0$.
Input: $n \geq 2/\varepsilon^2$.
Input: An untrusted advice $\bar{m} > 0$.
Output: A number $\tilde{\boldsymbol{m}}$.
Complexity: $O\left(\frac{\log n}{\varepsilon^{5/2}}(1 + \frac{m}{\bar{m}})\sqrt{n/\sqrt{\bar{m}}}\right)$.

1. If $\bar{m} \geq \frac{1}{4}\varepsilon n^2$:
   a. Set $k \leftarrow n - 1$.
   b. Run Estimate-LL-Edges-Advice$(n, \varepsilon, G, \bar{m}, k)$ and return the result as $\tilde{\boldsymbol{m}}$.
   c. Return $\tilde{\boldsymbol{m}}$.
2. Else:
   a. Set $k \leftarrow \frac{\sqrt{2}}{\sqrt{\varepsilon}} n^{\frac{1}{2}} \bar{m}^{\frac{1}{4}}$.
   b. Run Estimate-LL-Edges-Advice$(n, \varepsilon, G, \bar{m}, k)$ and let $\tilde{\boldsymbol{m}}_{LL}$ be the result.
   c. Run Estimate-L$_1$H-Edges-Advice$(n, \varepsilon, G, \bar{m}, k)$ and let $\tilde{\boldsymbol{m}}_{L_1H}$ be the result.
   d. Return $\tilde{\boldsymbol{m}} = \tilde{\boldsymbol{m}}_{LL} + \tilde{\boldsymbol{m}}_{L_1H}$.

Algorithm 8: Procedure Estimate-Edges-Advice-Low$(\varepsilon, G, \bar{m})$.

Input: A graph $G = (V, E)$ (V is explicitly given) accessible through Deg, Neigh, IS.
Input: An accuracy parameter $\varepsilon > 0$.
Input: An untrusted advice $\bar{m} \geq 1$.
Output: A number $\tilde{\boldsymbol{m}}$.
Complexity: $O(\frac{\log n}{\varepsilon^2}\sqrt{\bar{m}})$.

1. Set $k \leftarrow \min(\bar{m}, n-1)$.
2. If Small-$\bar{m}$-Guard$(G, \bar{m})$ accepts:
   a. Run Estimate-LL-Edges-Advice$(n, \frac{\varepsilon}{10}, G, \bar{m}, k)$ and return the result as $\tilde{\boldsymbol{m}}$.
3. Return $+\infty$.

Algorithm 9: Procedure Estimate-Edges-Iteration$(n, \varepsilon, G, \ell)$.

Input: $0 < \varepsilon \leq \frac{1}{15}$.
Input: G has n vertices (explicitly given) and $m \geq 1$ edges (unknown to the algorithm).
Complexity: $O(2^{\max(0,\ell-\ell_0)} \cdot 2^{\ell/4} \cdot \log n/\varepsilon^{5/2})$.

1. Let $\bar{m}_{\text{big}} \leftarrow \frac{n^2}{2^\ell}$, $\bar{m}_{\text{small}} \leftarrow 2^{\ell/2}$. (*cannot* assume that $\bar{m}_{\text{big}} \geq \bar{m}_{\text{small}}$)
2. Call Estimate-Edges-Advice-High$(n, \frac{1}{1000}\varepsilon, G, \bar{m}_{\text{big}})$ and let $\tilde{\boldsymbol{m}}_{\text{big}}$ be the result.
3. If $\frac{1}{4}\bar{m}_{\text{big}} \leq \tilde{\boldsymbol{m}}_{\text{big}} \leq \frac{4}{5}\bar{m}_{\text{big}}$:
   a. Return $\tilde{\boldsymbol{m}}_{\text{big}}$ as $\tilde{\boldsymbol{m}}$.
4. Call Estimate-Edges-Advice-Low$(n, \frac{1}{10}\varepsilon, G, \bar{m}_{\text{small}})$ and let $\tilde{\boldsymbol{m}}_{\text{small}}$ be the result.
5. If $\tilde{\boldsymbol{m}}_{\text{small}} \leq \frac{1}{\sqrt[4]{2}}\bar{m}_{\text{small}}$:
   a. Return $\tilde{\boldsymbol{m}}_{\text{small}}$ as $\tilde{\boldsymbol{m}}$.
6. Return REJECT.

▶ **Lemma 19.** *Assume that $\varepsilon \leq \frac{1}{15}$. If $\ell \leq \ell_0 - 2$, then with probability at least $1 - \frac{1}{50} \cdot 2^{(\ell-\ell_0)/2}$, Estimate-Edges-Iteration (Algorithm 9) either rejects or returns a number in the range $(1 \pm \varepsilon)m$. If $\ell = \ell_0 - 1$, then this probability is at least $1 - \frac{1}{100} \cdot 2^{(\ell-\ell_0)/2} - \frac{1}{100}$.*

▶ **Lemma 20.** *For $\varepsilon \leq \frac{1}{15}$, if $\ell = \ell_0$, then Estimate-Edges-Iteration (Algorithm 9) returns a number in the range $(1 \pm \varepsilon)m$ with probability at least $1 - \frac{1}{55}$.*

▶ **Lemma 21.** *Assume that $\varepsilon \leq \frac{1}{15}$. Considering Estimate-Edges-Bounded:*

- *If $\ell_{\max} \leq \ell_0 - 2$, then the probability of the ℓth call to Estimate-Edges-Iteration to return a non-reject value outside the range $(1 \pm \varepsilon)m$ is bounded by $\frac{2+\sqrt{2}}{50} \cdot 2^{(\ell-\ell_0)/2}$.*
- *If $\ell_{\max} = \ell_0 - 1$, then the probability to return a number outside the range $(1 \pm \varepsilon)m$ is bounded by $\frac{3}{50}$.*
- *If $\ell_{\max} \geq \ell_0$, then the probability to return a number in the range $(1 \pm \varepsilon)m$ is at least $1 - 3/34$.*

▶ **Lemma 22.** *For $\varepsilon \leq \frac{1}{15}$ and an input graph G with n vertices and m edges. Estimate-Edges (Algorithm 11) returns a number in the range $(1 \pm \varepsilon)m$ with probability at least $2/3$. Moreover, its expected query complexity bounded by $O\left(\frac{\log n}{\varepsilon^{5/2}} \min\left(\sqrt{m}, \sqrt{\frac{n}{\sqrt{m}}}\right)\right)$.*

Algorithm 10: Procedure Estimate-Edges-Bounded$(n, \varepsilon, G, \ell_{\max})$.

Input: $0 < \varepsilon \leq \frac{1}{10}$.
Input: G has n vertices (explicitly given) and m edges (unknown to the algorithm).
Input: $m \geq 1$.

1. For ℓ from 0 to $\ell_{\max}$:
   a. Cal Estimate-Edges-Iteration$(n, \varepsilon, G, \ell)$ and let $\tilde{\boldsymbol{m}}$ be its result.
   b. If $\tilde{\boldsymbol{m}} \neq$ REJECT:
      i. Return $\tilde{\boldsymbol{m}}$.
2. Return REJECT.

Algorithm 11: Procedure Estimate-Edges(n, ε, G).

Input: Integer n, parameter $\varepsilon \in (0, 1]$.
Input: Access to the oracles Deg, Neigh, IS of a graph $G = ([n], E)$.
Output: A number $\tilde{\boldsymbol{m}}$, an estimate for m.

1. If $\varepsilon > \frac{1}{15}$:
   a. Set $\varepsilon \leftarrow \frac{1}{15}$.
2. If IS$(V) = 1$:
   a. Return 0.
3. If $n < 2/\varepsilon^2$:
   a. For each $u \in V$:
      i. Let $d_u \leftarrow \deg(u)$ (via Deg).
   b. Return $\frac{1}{2}\sum_{u \in V} d_u$.
4. Let $\ell \leftarrow 0$.
5. While TRUE:
   a. Call Estimate-Edges-Bounded$(n, \varepsilon, G, \ell)$ and let $\tilde{\boldsymbol{m}}$ be its result.
   b. If $\tilde{\boldsymbol{m}} \neq$ REJECT:
      i. Return $\tilde{\boldsymbol{m}}$.
   c. $\ell \leftarrow \ell + 1$.

4 Lower Bound

In this section we prove the following lower-bound for edge estimation in the hybrid model. In particular, the lower bound follows from the following (stronger) statement.

▶ **Theorem 23.** *Consider any algorithm that is given query access to an unknown graph $G = (V, E)$ on n vertices and m edges via degree queries, neighbor queries and IS queries. Let $\varepsilon \in (0, 1)$ and define $R \stackrel{\text{def}}{=} \frac{1}{30} \min\left(\sqrt{m}, \sqrt{\frac{n}{\varepsilon\sqrt{m}}}\right)$. For any such algorithm whose expected number of queries is q (where q may depend on G), the probability that the algorithm outputs an estimate $\widehat{\boldsymbol{m}}$ satisfying $(1-\varepsilon)m \leq \widehat{\boldsymbol{m}} \leq (1+\varepsilon)m$ is at most $\frac{1}{2} + q/R$.*

Theorem 23 above bounds the success probability of any algorithm as a function of its expected number of queries. Rearranging this bound and applying Markov's inequality yields the following corollary for algorithms that succeed with probability at least 2/3.

▶ **Corollary 24.** *Let $\varepsilon \in (0,1)$. Every algorithm whose input is an access to a graph G of m edges (where m is unknown to the algorithm) through degree queries, neighbor queries and IS queries, whose output $\widehat{\boldsymbol{m}}$ satisfies $(1-\varepsilon)m \leq \widehat{\boldsymbol{m}} \leq (1+\varepsilon)m$ with probability at least $2/3$, must make $\Omega\left(\min\left(\sqrt{m}, \sqrt{\frac{n}{\varepsilon\sqrt{m}}}\right)\right)$ queries.*

The proof of Theorem 23 proceeds via a reduction from a specially designed string-testing problem to the problem of estimating the number of edges in a graph with a particular structure.

4.1 The string distinction problem

Let s be a string of length n over the alphabet $\Sigma = \{\mathrm{A}, \mathrm{K}, \mathrm{L}, \mathrm{H}\}$. We define a duality map as follows: A and K are the dual of themselves ($\overline{\mathrm{A}} = \mathrm{A}$, $\overline{\mathrm{K}} = \mathrm{K}$), and L and H are the dual of each other ($\overline{\mathrm{L}} = \mathrm{H}$, $\overline{\mathrm{H}} = \mathrm{L}$).

For $\sigma \in \Sigma$ and a string $s \in \Sigma^n$, let $n_\sigma(s) = |\{1 \leq i \leq n : s(i) = \sigma\}|$. For every $1 \leq i \leq n$, let

$$U_i \stackrel{\text{def}}{=} \begin{cases} \emptyset, & \text{if } s(i) = \mathrm{A}, \\ \{\, j \in [n] \setminus \{i\} : s(j) = \overline{s(i)} \,\}, & \text{otherwise.} \end{cases}$$

We assume that s is accessible through two oracles:

- **Full query.** Given an index $1 \leq i \leq n$, the oracle returns three entries:
  - The symbol $s(i)$.
  - The value $|U_i|$.
  - A uniformly random index from U_i, or $\perp$ if $U_i = \emptyset$.
- **Light query.** Given an index $1 \leq i \leq n$, the oracle returns the symbol $s(i)$.

We consider the task of distinguishing between strings of length n for which $n_{\mathrm{L}}(s) \cdot n_{\mathrm{H}}(s) = 0$ and strings for which $n_{\mathrm{L}}(s) \cdot n_{\mathrm{H}}(s) > \varepsilon\binom{n_{\mathrm{K}}(s)}{2}$. We say that an algorithm *distinguishes* between the above cases if one of the following occurs:

▶ **Definition 25** (Full-query distinguishing predicate)**.** *An algorithm performs a full query on a vertex i for which $s(i) \in \{\mathrm{L}, \mathrm{H}\}$ and obtained a non-$\perp$ index as the third entry of the result tuple.*

▶ **Definition 26** (Light-query distinguishing predicate)**.** *An algorithm performs a light query on a vertex i for which $s(i) = \mathrm{H}$ and another light query on a vertex j for which $s(j) = \mathrm{L}$.*

Observe that if either predicate occurs, the algorithm can deduce that $n_{\mathrm{L}}(s) \cdot n_{\mathrm{H}}(s) \neq 0$. Conversely, if neither predicate occurs, the algorithm cannot distinguish between the case where all $\{\mathrm{L}, \mathrm{H}\}$ symbols are the same (all L or all H) and the case where they are not.

4.2 Reduction to edge estimation

We can interpret a string s as a description of a graph of the following structure. A clique over the set $\{i : s(i) = \mathrm{K}\}$ and a biclique whose sides are $\{i : s(i) = \mathrm{L}\}$ and $\{i : s(i) = \mathrm{H}\}$. The number of edges is exactly $\binom{n_{\mathrm{K}}(s)}{2} + n_{\mathrm{L}}(s) \cdot n_{\mathrm{H}}(s)$. More formally,

▶ **Definition 27** (Reduction graph)**.** *Let $s \in \Sigma^n$ be a string. The* reduction graph *G_s is the graph with vertex set $\{1, \ldots, n\}$ defined as follows. An edge is placed between every pair of vertices i, j such that $s(i) = s(j) = \mathrm{K}$, and between every pair of vertices i, j such that*

$s(i) = \mathrm{L}$ *and* $s(j) = \mathrm{H}$. *Equivalently,* G_s *consists of a clique induced by the vertices labeled* K *and a complete bipartite graph between the vertices labeled* L *and those labeled* H. *Figure 1 illustrates the structure of* G_s.

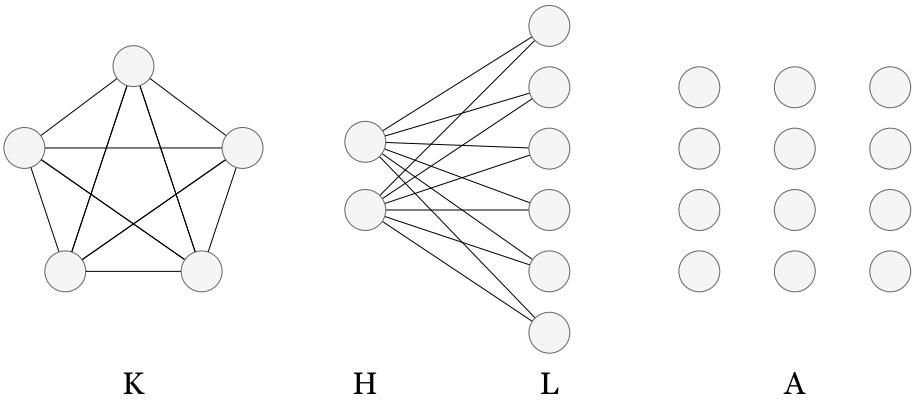


Figure 1 A diagram of the graph structure.

We describe a reduction that simulates graph queries using string queries. At the t-th step of the simulation, the following process is executed:

1. The t-th graph query, denoted Λ_t, is determined as a function of the outcomes of the preceding graph queries and the randomness of the algorithm.
2. The query Λ_t is translated into a sequence of string queries to the string s. Let $I_{t,1}, \ldots, I_{t,M_t}$ denote this sequence, where M_t is the number of string queries performed at step t. The value of M_t may depend on the outcomes of previous string queries, on the string s, and on the internal randomness of the translation algorithm.
3. Using the responses to the string queries $I_{t,1}, \ldots, I_{t,M_t}$, the simulation deterministically computes Y_t, the outcome of the t-th graph query.

At this point we describe the translation algorithm. Degree and neighbor queries are simulated using a single full string query, whereas an independent-set query is simulated by a randomized, adaptively determined sequence of light string queries.

Formally, the simulation proceeds as follows.

- **Degree query.** Given a vertex u we issue a full string query to s at position u and return the second component of the resulting triplet.
- **Neighbor query.** Given a vertex u we issue a full string query to s at position u and return the third component of the resulting triplet.
- **Independent-set query.** Given a vertex set S we first apply a uniformly random permutation to the vertices of S. We then process the vertices in this order, issuing light string queries, until one of the following stopping conditions is met:
  1. Two vertices labeled K are revealed, certifying that S is not an independent set.
  2. One vertex labeled L and one vertex labeled H are revealed, certifying that S is not an independent set.
  3. All vertices in S have been processed without triggering either of the above conditions, in which case S is declared an independent set.

▶ Remark 28. In the simulation of an independent-set query, whenever the symbol of a vertex has already been revealed, either by a previous string query or as part of the output of another query, we reuse this information and do not issue an additional query for that vertex.

The correctness of the simulation is trivial by the construction details. In the following we bound the simulation complexity. For local queries this is directly implied from the implementation:

▶ **Observation 29.** *The simulation of any local query (degree or neighbor) incurs exactly one full string query.*

It remains to bound the expected number of light string queries used in the simulation of an independent-set query.

▶ **Lemma 30.** *A vertex is said to be* involved *in a string query if it is either (i) directly queried, or (ii) appears in the output of that query. Every simulated graph query involves at most two vertices u such that $s(u) = \mathrm{K}$.*

Proof. We consider the two types of simulated graph queries.

For a local query (degree or neighbor), the simulation performs a single full string query. Such a query involves the queried vertex and at most one additional vertex appearing as part of the output (as the third entry). Hence, the simulation of a local query involves at most two vertices in total, and therefore at most two vertices labeled K.

For an independent-set query, the simulation issues a sequence of light string queries, each involving exactly one vertex (the queried vertex). By construction, the simulation halts immediately once it has revealed two vertices u with $s(u) = \mathrm{K}$. Consequently, throughout the execution of the simulation, at most two K-vertices can be involved in the string query. ◀

▶ **Lemma 31.** *Assume that $\boldsymbol{s} \in \{\mathrm{K}, \mathrm{A}\}^n$ is drawn from a distribution that is invariant under permutations of its coordinates; that is, all permutations of $\boldsymbol{s}$ have equal probability (while strings that are not permutations of one another may have different probabilities). Suppose that the t-th query issued by the graph algorithm is an independent-set query. If $t < \frac{1}{2} n_{\mathrm{K}}(\boldsymbol{s})$, then conditioned on any outcomes of the first $t-1$ queries, the simulation of the t-th query performs at most $\frac{2n}{n_{\mathrm{K}}(\boldsymbol{s}) - 2t}$ light string queries in expectation.*

4.3 A hard to distinguish construction

To construct a pair of hard-to-distinguish graphs, we define a few parameters. For given ε, n and n_k, let $d = n_k/2n$ (so that $\sqrt{2nd} = n_k$). Based on n, n_k, we define n_h and n_ℓ as following:

- $n_h = \left\lceil \sqrt{\varepsilon} n^{1/4} d^{3/4} \right\rceil$.
- $n_\ell = \left\lceil \sqrt{\varepsilon} n^{3/4} d^{1/4} \right\rceil$ if $n_h \geq 2$, and otherwise $n_\ell = \lceil \varepsilon n d \rceil$.

We state the following simple observation.

▶ **Observation 32.** *Let n_ℓ and n_h be defined as above. Then,*

1. *$n_h = 1$ if and only if $d \leq 1/(\varepsilon^{2/3} n^{1/3})$.*
2. *If $n_h = 1$, then $n_\ell + n_h \leq 3n/\sqrt{nd}$.*
3. *If $n_h \geq 2$, then $n_\ell + n_h \leq 4\sqrt{\varepsilon}\, n^{3/4} d^{1/4}$.*

We now define a distribution over hard-to-distinguish pairs of strings.

▶ **Definition 33** (Hard-to-distinguish strings)**.** *Let n_k, n_h, and n_ℓ be given. We draw a pair of strings $(\boldsymbol{s}_1, \boldsymbol{s}_2)$ as follows:*

- *Choose a set $\mathbf{K} \subseteq \{1, \ldots, n\}$ of size n_k uniformly at random.*
- *Define $\mathbf{A}_1 = \{1, \ldots, n\} \setminus \mathbf{K}$.*

- *Define* $\boldsymbol{s}_1$ *by* $\boldsymbol{s}_1(i) = \begin{cases} \text{K} & \text{if } i \in \mathbf{K}, \\ \text{A}, & \text{if } i \in \mathbf{A}_1. \end{cases}$
- *Choose a set* $\mathbf{H}_2 \subseteq \mathbf{A}_1$ *of size* n_h *uniformly at random.*
- *Choose a set* $\mathbf{L}_2 \subseteq (\mathbf{A}_1 \setminus \mathbf{H}_2)$ *of size* n_ℓ *uniformly at random.*
- *Define* $\mathbf{A}_2 = \mathbf{A}_1 \setminus (\mathbf{H}_2 \cup \mathbf{L}_2)$.
- *Define* $\boldsymbol{s}_2$ *by* $\boldsymbol{s}_2(i) = \begin{cases} \text{K} & \text{if } i \in \mathbf{K}, \\ \text{H} & \text{if } i \in \mathbf{H}_2, \\ \text{L}, & \text{if } i \in \mathbf{L}_2, \\ \text{A}, & \text{if } i \in \mathbf{A}_2. \end{cases}$

A pair of hard-to-distinguish graphs is obtained by sampling $(\boldsymbol{s}_1, \boldsymbol{s}_2)$ according to this distribution and defining the corresponding graphs $G_{\boldsymbol{s}_1}$ and $G_{\boldsymbol{s}_2}$.

▶ **Lemma 34.** *For given* $0 < \varepsilon \leq \frac{1}{11}$, $n \geq 16$ *and* $1 \leq n_k \leq \frac{1}{2}n$, *let* $R = \frac{1}{20}\min\left(n_k, \sqrt{\frac{n}{\varepsilon n_k}}\right)$. *Assume that we draw a hard-to-distinguish pair* $(\boldsymbol{s}_1, \boldsymbol{s}_2)$ *based on the parameters* ε, n *and* n_k. *Then, for any randomized algorithm making at most* q *graph queries in expectation, the probability that the sequences of query answers produced on inputs* $G_{\boldsymbol{s}_1}$ *and* $G_{\boldsymbol{s}_2}$ *differ is at most* q/R.

We can now finish with the proof of the lower bound.

Proof of Theorem 23. Consider a randomized algorithm $\mathcal{A}$ that makes at most q queries.

We draw a $(\boldsymbol{s}_1, \boldsymbol{s}_2)$ according to the parameters $\varepsilon' = 3\varepsilon$, n and $n_k = \lfloor\sqrt{2m}\rfloor$. Note that for $m \geq 6$ we have that $\sqrt{m} \leq n_k \leq 2\sqrt{m}$.

By Lemma 34, the query answer sequences of $\mathcal{A}$ on $G_{\boldsymbol{s}_1}$ and $G_{\boldsymbol{s}_2}$ are $20q/\min(n_k, \sqrt{n/(\varepsilon n_k)})$-close. Consequently, the probability that $\mathcal{A}$ distinguishes $G_{\boldsymbol{s}_1}$ from $G_{\boldsymbol{s}_2}$ is bounded by

$$\mathbf{Pr}[\mathcal{A}(G_{\boldsymbol{s}_1}) \neq \mathcal{A}(G_{\boldsymbol{s}_2})] \leq \frac{1}{2} + \frac{20q}{\min\left(n_k, \sqrt{\frac{n}{\varepsilon n_k}}\right)} \leq \frac{1}{2} + \frac{20q}{\min\left(\sqrt{m}, \sqrt{\frac{n}{\varepsilon\sqrt{2m}}}\right)} < \frac{1}{2} + \frac{q}{R}.$$

Let m_1, m_2 be the number of edges in $G_{\boldsymbol{s}_1}$ and $G_{\boldsymbol{s}_2}$ respectively. Observe that $m_1 = \binom{n_k}{2} \leq \frac{1}{2}(n_k)^2 \leq m$ and that $m_2 \geq m_1 + \varepsilon' m \geq (1+\varepsilon')m_1 = (1+3\varepsilon)m_1$. If $\mathcal{A}$ would estimate the number of edges in G within $(1 \pm \varepsilon)$-multiplicative error with probability at least $\frac{1}{2} + q/R$, then it would be a contradiction, since $(1+\varepsilon)m_1 < (1-\varepsilon)(1+3\varepsilon)m_1 \leq (1-\varepsilon)m_2$. ◀

References

1. Hasan Abasi and Bshouty Nader. On learning graphs with edge-detecting queries. In *Algorithmic Learning Theory*, pages 3–30. PMLR, 2019. URL: `http://proceedings.mlr.press/v98/abasi19a.html`.
2. Raghavendra Addanki, Andrew McGregor, and Cameron Musco. Non-adaptive edge counting and sampling via bipartite independent set queries. *arXiv preprint*, 2022. `doi:10.48550/arXiv.2207.02817`.
3. Dana Angluin and Jiang Chen. Learning a hidden graph using $O(\log n)$ queries per edge. *Journal of Computer and System Sciences*, 74(4):546–556, 2008. `doi:10.1016/J.JCSS.2007.06.006`.
4. Dana Angluin, Jiang Chen, and Manfred Warmuth. Learning a hidden hypergraph. *Journal of Machine Learning Research*, 7(79):2215–2236, 2006. URL: `http://jmlr.org/papers/v7/angluin06a.html`.
5. Sepehr Assadi, Michael Kapralov, and Sanjeev Khanna. A simple sublinear-time algorithm for counting arbitrary subgraphs via edge sampling. In *Proceedings of the 2019 ACM Conference on Innovations in Theoretical Computer Science (ITCS '2019)*, 2019.

6 Paul Beame, Sariel Har-Peled, Sivaramakrishnan Natarajan Ramamoorthy, Cyrus Rashtchian, and Makrand Sinha. Edge estimation with independent set oracles. In *Proceedings of the 2018 ACM Conference on Innovations in Theoretical Computer Science (ITCS '2018)*, 2018.

7 Lorenzo Beretta, Deeparnab Chakrabarty, and C Seshadhri. Faster estimation of the average degree of a graph using random edges and structural queries. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 939–971. SIAM, 2026. doi:10.1137/1.9781611978971.38.

8 Lorenzo Beretta and Jakub Tětek. Better sum estimation via weighted sampling. *ACM Transactions on Algorithms*, 20(3):1–33, 2024. URL: https://doi.org/10.1145/3650030.

9 Anup Bhattacharya, Arijit Bishnu, Arijit Ghosh, and Gopinath Mishra. Hyperedge estimation using polylogarithmic subset queries. *arXiv preprint*, 2019. arXiv:1908.04196.

10 Anup Bhattacharya, Arijit Bishnu, Arijit Ghosh, and Gopinath Mishra. On triangle estimation using tripartite independent set queries. *Theory of Computing Systems*, 65(8):1165–1192, 2021. doi:10.1007/S00224-021-10043-Y.

11 Anup Bhattacharya, Arijit Bishnu, Arijit Ghosh, and Gopinath Mishra. Faster counting and sampling algorithms using colorful decision oracle. *TOCT*, 16(2):1–19, 2024. doi:10.1145/3657605.

12 Arijit Bishnu, Debarshi Chanda, and Gopinath Mishra. Arboricity and random edge queries matter for triangle counting using sublinear queries. *arXiv preprint*, 2025. doi:10.48550/arXiv.2502.15379.

13 Arijit Bishnu, Arijit Ghosh, Gopinath Mishra, and Manaswi Paraashar. Efficiently sampling and estimating from substructures using linear algebraic queries. *arXiv preprint*, 2019. arXiv:1906.07398.

14 Chao L Chen and William H Swallow. Using group testing to estimate a proportion, and to test the binomial model. *Biometrics*, pages 1035–1046, 1990.

15 Xi Chen, Amit Levi, and Erik Waingarten. Nearly optimal edge estimation with independent set queries. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2916–2935. SIAM, 2020. doi:10.1137/1.9781611975994.177.

16 Holger Dell and John Lapinskas. Fine-grained reductions from approximate counting to decision. *TOCT*, 13(2):1–24, 2021. doi:10.1145/3442352.

17 Holger Dell, John Lapinskas, and Kitty Meeks. Approximately counting and sampling small witnesses using a colorful decision oracle. *SIAM Journal on Computing*, 51(4):849–899, 2022. doi:10.1137/19M130604X.

18 Robert Dorfman. The detection of defective members of large populations. *The Annals of mathematical statistics*, 14(4):436–440, 1943.

19 Talya Eden, Dana Ron, and C Seshadhri. Sublinear time estimation of degree distribution moments: The arboricity connection. *SIAM Journal on Discrete Mathematics*, 33(4):2267–2285, 2019. doi:10.1137/17M1159014.

20 Talya Eden and Will Rosenbaum. On sampling edges almost uniformly. In *Proceedings of the 1st Symposium on Simplicity in Algorithms (SOSA)*, pages 7–1, 2018.

21 Uriel Feige. On sums of independent random variables with unbounded variance and estimating the average degree in a graph. *SIAM Journal on Computing*, 35(4):964–984, 2006. doi:10.1137/S0097539704447304.

22 Oded Goldreich and Dana Ron. Approximating average parameters of graphs. *Random Structures & Algorithms*, 32(4):473–493, 2008. doi:10.1002/RSA.20203.

23 Amit Levi, Ramesh Krishnan S Pallavoor, Sofya Raskhodnikova, and Nithin Varma. Erasure-resilient sublinear-time graph algorithms. *TOCT*, 14(1):1–22, 2021. doi:10.1145/3488250.

24 Dana Ron and Gilad Tsur. The power of an example: Hidden set size approximation using group queries and conditional sampling. *ACM Trans. Comput. Theory*, 8(4):1–19, 2016. URL: https://doi.org/10.1145/2930657.

25 Larry Stockmeyer. The complexity of approximate counting. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing (STOC)*, pages 118–126, 1983.

26 Larry Stockmeyer. On approximation algorithms for $\#P$. *SIAM Journal on Computing*, 14(4):849–861, 1985. doi:10.1137/0214060.

27 William H Swallow. Group testing for estimating infection rates and probabilities of disease transmission. *Phytopathology (USA)*, 1985.

28 Jakub Tětek and Mikkel Thorup. Edge sampling and graph parameter estimation via vertex neighborhood accesses. In *Proceedings of the 54th annual ACM SIGACT symposium on theory of computing (STOC)*, pages 1116–1129, 2022.

29 Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity. In *18th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 222–227. IEEE Computer Society, 1977.

Two-Sided Lossless Expanders in the Unbalanced Setting

Eshan Chattopadhyay
Cornell University, Ithaca, NY, USA

Mohit Gurumukhani
Cornell University, Ithaca, NY, USA

Noam Ringach
Cornell University, Ithaca, NY, USA

Yunya Zhao
Cornell University, Ithaca, NY, USA

Abstract

We present the first explicit construction of *two-sided* lossless expanders in the unbalanced setting (bipartite graphs that have polynomially many more nodes on the left than on the right). Prior to our work, all known explicit constructions in the unbalanced setting achieved only one-sided lossless expansion.

Specifically, we show that the one-sided lossless expanders constructed by Kalev and Ta-Shma (RANDOM'22) – that are based on multiplicity codes introduced by Kopparty, Saraf, and Yekhanin (STOC'11) – are, in fact, two-sided lossless expanders. Moreover, we show that our result is tight, thus completely characterizing the graph of Kalev and Ta-Shma.

2012 ACM Subject Classification Theory of computation → Expander graphs and randomness extractors

Keywords and phrases Pseudorandomness, lossless expanders, multiplicity codes, condensers

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.34

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2409.04549`

Funding *Eshan Chattopadhyay*: Supported by a Sloan Research Fellowship, NSF CAREER Award 2045576 and NSF Award CCF-2514586.
Mohit Gurumukhani: Supported by a Sloan Research Fellowship, NSF CAREER Award 2045576 and NSF Award CCF-2514586.
Noam Ringach: Supported by NSF GRFP grant DGE – 2139899, NSF CAREER Award 2045576, a Sloan Research Fellowship and NSF Award CCF-2514586.
Yunya Zhao: Supported by a Sloan Research Fellowship, NSF CAREER Award 2045576 and NSF Award CCF-2514586.

Acknowledgements We would like to thank Itay Kalev for helpful feedback on our paper. We are also grateful to insightful suggestions from anonymous reviewers that led to simplifying some of our proofs.

1 Introduction

Lossless expanders are graphs in which small sets of vertices have almost as many neighbors as possible. Formally, we say that a d-regular graph $G = (V, E)$ is a (K, A)-*expander* if for all sets $S \subseteq V$ of size at most K we have that $|\Gamma(S)| \geq A\,|S|$ where $\Gamma(S)$ is the neighborhood of S. Generally, we desire that K is as large as possible with $K = \Omega(|V|/d)$. When $A = (1-\varepsilon)d$

© Eshan Chattopadhyay, Mohit Gurumukhani, Noam Ringach, and Yunya Zhao;
licensed under Creative Commons License CC-BY 4.0

for some small ε, we say that G is a (K, ε)-*lossless expander* since only a small fraction of the total number of possible neighbors is lost. It is well-known that a random d-regular graph is a $(K = \gamma n, \varepsilon = 0.01)$-lossless expander with high probability, for some constant $\gamma > 0$.

A reasonable question after seeing this definition is whether other notions of expansion, such as spectral or edge expansion, can be used to derive such graphs. Unfortunately, while Ramanujan graphs (optimal spectral expanders) do have expansion factor arbitrarily close to $A = d/2$, there also exist examples of Ramanujan graphs with expansion factor exactly $A = d/2$, showing that spectral expansion does not necessarily give rise to lossless expansion [14].

The study of lossless expanders has paid special attention to bipartite graphs due to their connection with randomness condensers. A *one-sided lossless* expander is a bipartite graph $G = (L \sqcup R, E)$ where every "small enough" set on the left expands losslessly to the right. It is standard to view lossless condensers and one-sided bipartite lossless expanders as related objects. For this purpose, it is natural to talk about highly unbalanced bipartite graphs in which $|L| \gg |R|$, that is, the neighbor function that takes in a left vertex and the index of a neighbor and outputs the right vertex has a much shorter output length than input length. Current explicit constructions for unbalanced one-sided lossless expanders [21, 22, 10, 15] – with the best parameters achieved by [10, 15] [1] – have found a wide array of applications in coding theory [20], extractor constructions [21, 22, 10, 7], derandomization [5],[2] and probabilistic data structures [23, 1], with the unbalanced nature of the graph being essential.

We say a bipartite graph is a *two-sided lossless* expander if lossless expansion happens in both directions. Intuitively, in the unbalanced case where $|L| \gg |R|$, lossless expansion is hard to achieve from left to right because there is little room on the smaller side to allow for expansion from the much larger side; on the other hand, the right-to-left expansion seems to be much easier at first sight for the same reason. However, despite aforementioned constructions for one-sided lossless expanders as well as their broad applications, there has been no known explicit construction of unbalanced two-sided lossless expanders before this work. Therefore, it is an extremely intriguing direction to further the theory of lossless expanders. We are not aware of any existing applications of two-sided lossless expanders with a polynomial imbalance between the left and right – we leave this as an interesting open problem – but we believe that a deeper understanding of the structure of these objects is the first step towards making connections with other topics and finding new applications.

In this work, we try to fill the gap, namely, the lack of explicit unbalanced two-sided lossless expanders by closely studying the bipartite graph of [15] based on the multiplicity codes of [16]. [3] For simplicity, we refer to this graph as the "KT graph" (see Definition 20). As our main result, we show that the KT graph is a two-sided lossless expander. Moreover, we prove that the expansion is tight up to a constant multiplicative factor.

We note that lossless expanders have been extensively studied in the *balanced* setting as well. We discuss this in Section 1.2.

1.1 Our results

We first define a two-sided lossless expander formally:

[1] The lossless expanders of [15] have slightly better dependence on constants compared to [10].
[2] [5] instantiated Goldreich's PRG [8] with the lossless expander of [15].
[3] It is natural to consider whether the unbalanced expander from [10] is a two-sided lossless expander. It will be interesting to determine this since the GUV graph is not even right-regular – see Appendix B for details.

▶ **Definition 1** (Two-sided lossless expander). *We say that a (D_L, D_R)-regular bipartite graph $G = (L \sqcup R, E)$ is a* two-sided (K_L, A_L, K_R, A_R)-expander *if for any subset $S_L \subseteq L$ such that $|S_L| \leq K_L$ we have that $|\Gamma_{\rightarrow}(S_L)| \geq A_L |S_L|$ and similarly that for any subset $S_R \subseteq R$ such that $|S_R| \leq K_R$ we have that $|\Gamma_{\leftarrow}(S_R)| \geq A_R |S_R|$. When $A_L = (1-\varepsilon_L)D_L$ and $A_R = (1-\varepsilon_R)D_R$ for small $\varepsilon_L, \varepsilon_R > 0$, we say that G is a* two-sided $(K_L, \varepsilon_L, K_R, \varepsilon_R)$-lossless expander.

With the above definition, we are ready to state the main theorems:

▶ **Theorem 2** (Informal version of Theorem 30, bipartite two-sided lossless expander). *For infinitely many N and all constant $0 < \delta \leq 0.99$, there exists an explicit, biregular, two-sided $(K_L, \varepsilon_L = 0.01, K_R, \varepsilon_R = 0.01)$ lossless expander $\Gamma_{\rightarrow} : [N] \times [D_L] \to [M]$ where $D_L = \text{poly}(\log N)$, $N^{1.01\delta - o(1)} \leq M \leq D_L \cdot N^{1.01\delta}$, $K_L = N^{\delta}$, and finally $K_R = \min(O(M/D_L), O(N/(MD_L)))$.*

On the other hand, for any of the explicit graphs constructed here, there exists a set $S \subseteq R$, $|S| = O(K_R)$, on the right side which has less than $(1 - \varepsilon_R/2) \cdot D_R \cdot |S|$ neighbors on the left.

▶ Remark 3. Because [15] has optimal left degree of their bipartite graph (up to polynomial factors), we achieve optimal left-degree as well and, with respect to this, achieve optimal right degree, optimal expansion constant, optimal size of sets of vertices on left side that losslessly expand, and optimal size of sets of vertices on right side that losslessly expand when $M \leq \sqrt{N}$. [4]

▶ Remark 4. When $\delta \leq 0.49$, the value of K is almost optimal (a trivial upper bound is $K \leq N/D$) since in that regime, $K = N^{\delta} \geq (N/D)^{0.99}$.

1.2 Lossless expanders in the balanced setting

A parallel line of work considers *balanced* bipartite lossless expanders, with motivations from coding theory. There have been explicit constructions of balanced one-sided lossless expanders [2, 4, 9], with which one can construct good error correcting codes [20]. There has also been progress in understanding two-sided lossless expanders in the balanced setting. Lossless expansion was shown to be feasible in high-girth regular graphs [18, 13], taking the bipartite double cover of which implies a balanced two-sided lossless expander. Chen [3] achieved balanced two-sided lossless expansion for polynomial-sized sets as one of their several results; however, their size of expanding sets is not optimal in the balanced setting. Our analysis proves optimal expanding set size in the unbalanced setting (see Remark 3).

It was shown in [17] that balanced two-sided lossless expanders with constant degree, constant imbalance, and certain algebraic properties have applications to good quantum low-density parity check (qLDPC) codes. A line of recent works [13, 11, 12] recently explicitly constructed these two-sided lossless expanders with the required algebraic property and successfully instantiated the qLDPC construction of [17]. Our work in the polynomially unbalanced setting is incomparable to the balanced setting, as it is possible to achieve constant degree in the balanced setting, while having a polynomial imbalance in left and right nodes forces the graph to have non-constant degrees.

[4] To see that this setting of K_R is indeed optimal, note that in a (D_L, D_R)-biregular graph it must be that $N \cdot D_L = M \cdot D_R$ and so $\frac{M}{D_L} = \frac{N}{D_R}$. Hence, $K_R = O(M/D_L) = O(N/D_R)$, the largest possible size.

2 Proof Overview

In this section, we outline the proof of Theorem 2 – our two-sided lossless expander.

2.1 Two-sided lossless expander

We show that the bipartite graph defined in [15] based on multiplicity codes is a two-sided lossless expander. The left-to-right lossless expansion was shown in [15]. Our main contribution is showing that the KT graph also expands losslessly from right to left. To do this, we first show that the KT graph is right-regular. Second, for any pair of right vertices, we compute the exact number of common left neighbors they have. Finally, for any not-too-large subset on the right, we lower bound the number of its left neighbors by using the inclusion-exclusion principle to subtract all possible double counted common left neighbors from the total number of outgoing edges.

We state an informal version of our result and present details on the strategy sketched above.

▶ **Theorem 5** (Informal version of Theorem 23). *For every field $\mathbb{F}_q$ and $n, s \in \mathbb{N}$ with $15 \leq (s+1) < n < char(\mathbb{F}_q)$, and any $\delta > 0$, there exists an explicit bipartite graph $G = (L \sqcup R, E)$ with $L = \mathbb{F}_q^n, R = \mathbb{F}_q^{s+2}$ with left degree $d_L = q$ and right degree $d_R = q^{n-(s+1)}$ such that G is a two-sided (K_L, A_L, K_R, A_R) expander where $K_L = \Omega(q^{s+1})$, $A_L = q - n(s+2)$, $K_R = \delta q^{\min(s+1, n-(s+1))}$, $A_R = \left(1 - O\left(\delta \cdot \frac{q-1}{q}\right)\right) q^{n-(s+1)}$.*

Theorem 2 is obtained from Theorem 5 by instantiating the parameters appropriately (see Section 4.3 for more details). We now define the KT graph and then claim that it is a lossless right expander.

▶ **Definition 6** (The KT graph [15]). *Let $q, n, s \in \mathbb{N}$ be such that q is a prime power, characteristic of the finite field $\mathbb{F}_q \geq n$ and $s + 2 \leq n$. We define $G = (L \sqcup R, E)$ where $L = \mathbb{F}_q^n, R = \mathbb{F}_q^{s+2}$. The left degree is q and for any $f \in \mathbb{F}_q^n$ and $y \in \mathbb{F}_q$, the y'th neighbor of f is defined as follows: Identify f as member of $\mathbb{F}_q[X]$ with degree of f at most $n-1$; then, the neighbor $\Gamma_{\rightarrow}(f, y)$ will be $(y, f^{(0)}(y), \ldots, f^{(s)}(y))$ where $f^{(i)}$ is the i'th iterative derivative of f.*[5]

▶ **Theorem 7** (The KT graph losslessly expands from the right). *The KT graph G is a right (K_R, A_R)-lossless expander where $K_R = \delta \min(|R|, |L|/|R|)$, $\varepsilon_R = O(\delta \cdot \frac{q-1}{q})$ for arbitrary $0 < \delta < 1$. In other words, for any subset $T \subseteq R$, $|T| \leq K_R$, T has at least $(1 - \varepsilon_R) d_R |T|$ neighbors on the left.*

Theorem 5 immediately follows from left expansion shown by [15] and Theorem 7. For the rest of this section, we focus on proving Theorem 7 that relies on the following two key lemmas.

▶ **Lemma 8** (Right regularity). *The KT graph G is right-regular and has right-degree $d_R = q^{n-(s+1)}$.*

▶ **Lemma 9** (Number of common left neighbors). *For any pair of right-vertices $w_1, w_2 \in \mathbb{F}_q^{s+2}$ such that $w_1 = (y_1, z_1), w_2 = (y_2, z_2)$ where $y_1 \neq y_2 \in \mathbb{F}_q$ and $z_1, z_2 \in \mathbb{F}_q^{s+1}$, we have $|\Gamma_{\leftarrow}(y_1, z_1) \cap \Gamma_{\leftarrow}(y_2, z_2)| = q^{n-(2s+2)}$ if $n \geq 2s+2$ and $|\Gamma_{\leftarrow}(y_1, z_1) \cap \Gamma_{\leftarrow}(y_2, z_2)| \leq 1$ if $n \leq 2s+2$.*

[5] Recall that the i-th iterative derivative of f is simply the the standard derivatve $\frac{\mathrm{d}}{\mathrm{d}x}$ applied i times to f.

Theorem 7 then follows by an application of the inclusion-exclusion principle – subtracting the maximum number of common neighbors between any pair of vertices in T from the total number of edges leaving T – we get the required lower bound on the size of T's left neighborhood.

We now discuss the proof techniques for showing Lemma 8 and Lemma 9. We start by making a simple but useful observation on the structure of the KT graph G.

▶ **Observation 10.** *Fix $w = (y, z_0, \cdots, z_s) \in R$ and let $f \in L$ be any left-neighbor of w. Then it must be the case that w is the y'th neighbor of f. Now for any $w' \in R$ such that $w' = (y, z'_0, \cdots, z'_s)$, it holds that $f \notin \Gamma_{\leftarrow}(w')$. This is saying that any pair of right vertices (w, w') that come from the same seed* [6] *must have disjoint left neighborhoods.*

Central to our analysis of the right degree and the number of common left neighbors are the following linear maps.

▶ **Definition 11.** *For $y \in \mathbb{F}_q$, define the map $\psi_y(f) : \mathbb{F}_q^n \to \mathbb{F}_q^{s+1}$ as follows: Interpret $f \in \mathbb{F}_q[X]$ as a degree $\leq n-1$ polynomial and map it to $(f^{(0)}(y), \ldots, f^{(s)}(y))$ where $f^{(i)}$ is the i'th iterative derivative of f.*

We note that ψ_y is a $\mathbb{F}_q$-linear map, for any $y \in \mathbb{F}_q$.

▶ **Definition 12.** *For $y_1, y_2 \in \mathbb{F}_q$, $y_1 \neq y_2$, define the map $\psi_{y_1,y_2}(f) : \mathbb{F}_q^n \to \mathbb{F}_q^{2(s+1)}$ as the concatenation of the respective linear maps, that is, $\psi_{y_1,y_2}(f) = (\psi_{y_1}(f), \psi_{y_2}(f))$.*

Proving the above lemmas (about the KT graph) now boils down to analyzing both ψ_y and ψ_{y_1,y_2} for all $y, y_1, y_2 \in \mathbb{F}_q$.

1. We show that ψ_y for $y \in \mathbb{F}_q$ is surjective, which along with Observation 10 implies Lemma 8: For any $w = (y, z_0, \cdots, z_s) \in R$, the set of its left neighbors is $\{f \in L \mid (y, \psi_y(f)) = w\} = \psi_y^{-1}(z_0, \cdots, z_s)$. Therefore, the right degree $D_R = \left|\psi_y^{-1}(z_0, \cdots, z_s)\right| = q^n/q^{s+1} = q^{n-(s+1)}$.
2. We show ψ_{y_1,y_2} is surjective when $n \geq 2s+2$ and injective $n \leq 2s+2$, which implies Lemma 9: Similar to above, let $w_1 = (y_1, z_1) \in R$ and $w_2 = (y_2, z_2) \in R$, $y_1 \neq y_2$, be any pair of right vertices from *different* seeds. We extend Observation 10 to see that the number of $f \in L$ such that $(y_1, \psi_{y_1}(f)) = w_1$ and $(y_2, \psi_{y_2}(f)) = w_2$ is exactly $\left|\psi_{y_1,y_2}^{-1}(z_1, z_2)\right|$. When $n \geq 2s+2$, this map is surjective, and the number of left neighbors shared by w_1 and w_2 is $q^{n-(2s+2)}$. When $n \leq 2s+2$, this map is injective and the number of shared neighbors is at most 1.

To conclude the proof, we carry out Hermite interpolation (see Lemma 27) by applying the Chinese remainder theorem to show the surjectivity and injectivity of the maps ψ_y, ψ_{y_1} and ψ_{y_2}.

2.1.1 Tightness of right expansion

We will show that our parameters from Theorem 7 are tight. Throughout this section, we let $P_d \subseteq \mathbb{F}_q[x]$ be the polynomials of degree exactly d and $P_{<d} \subseteq \mathbb{F}_q[x]$ be the vector space over $\mathbb{F}_q$ of polynomials of degree strictly less than d. First, notice that when $|R| > |L|/|R|$, the right expansion is optimal by Remark 3. Hence, we only focus on the case when $s+1 < n < 2s+2$ and show the following:

[6] We sometimes refer to $y \in \mathbb{F}_q$ as the "seed", like in the condensers literature.

▶ **Theorem 13** (Informal version of Theorem 32). *For $s+1 < n < 2s+2$ and any $0 < \delta \leq 2$, there exists $T \subseteq R$ such that $|T| = \delta q^{n-s-1}$ and $|\Gamma_{\leftarrow}(T)| = (1-\varepsilon)D_R|T|$ with $\varepsilon = \delta/4$.*

For fixed $y_1 \neq y_2 \in \mathbb{F}_q$, we will construct $T = T_1 \sqcup T_2$ satisfying the following:

- $|T_1|, |T_2| = (\delta/2)q^{n-s-1}$.
- The first coordinate of every element of T_1 is always y_1 and of T_2 is always y_2.
- For all $t_1 \in T_1$ and $t_2 \in T_2$, there exists a degree less than n polynomial f such that $\Gamma_{\rightarrow}(f, y_1) = t_1$ and $\Gamma_{\rightarrow}(f, y_2) = t_2$.

We then observe that $|\Gamma_{\leftarrow}(T)| = |\Gamma_{\leftarrow}(T_1)| + |\Gamma_{\leftarrow}(T_2)| - |T_1| \cdot |T_2|$. Since the right degree of G is q^{n-s-1}, we have that $|\Gamma_{\leftarrow}(T_1)| = |\Gamma_{\leftarrow}(T_2)| = (\delta/2)q^{2n-2s-2}$. Hence, we compute that $|\Gamma_{\leftarrow}(T)| = (\delta - \delta^2/4)q^{2n-2s-2} = (1-\delta/4)D_R|T|$, proving Theorem 13.

To construct such T_1, T_2, it suffices to construct $S_1, S_2 \subset \mathbb{F}_q^{s+1}$ with $|S_1| = |S_2| = K/2$ such that $S_1 \times S_2 \subset \psi_{y_1,y_2}(P_{<n})$ (see Lemma 35 for a formal claim). Indeed, we can let $T_1 = (y_1, S_1)$ and $T_2 = (y_2, S_2)$ and check that T_1 and T_2 have the desired properties.

Before we show how to construct such S_1 and S_2, we will need to introduce a few algebraic objects. Let $g_1(x) = (x-y_1)^{s+1}, g_2(x) = (x-y_2)^{s+1}$, and let $\varphi : \mathbb{F}_q[x] \to \mathbb{F}_q[x]/g_1 \times \mathbb{F}_q[x]/g_2$ as $\varphi(f) = (f \mod g_1, f \mod g_2)$.

We then prove a structural result regarding φ:

▶ **Lemma 14** (Informal version of Lemma 41). *For $s+1 < d < 2s+2$, we have*

$$\varphi(P_{<d}) = \bigcup_{h \in P_{<d-(s+1)}} \{(f, f+\sigma(h)) \mid f \in P_{\leq s}\}.$$

Here, $\sigma : P_{\leq n-(s+1)} \to P_{\leq s}$ is a specially chosen injective homomorphism that we define later.

To construct such S_1 and S_2, we construct $R_1 \subset \mathbb{F}_q[x]/g_1, R_2 \subset F_q[x]/g_2$ such that $R_1 \times R_2 \subset \varphi(P_{<n})$ (see Lemma 39 for formal claim). Once we have such R_1, R_2, we let $S_1 = \psi_{y_1}(R_1)$ and $S_2 = \psi_{y_2}(R_2)$. We then check that if $f \in P_{<n}$ is such that $\varphi(f) = (r_1, r_2)$, then $f(y_1) = r_1(y_1) = s_1$ and $f(y_2) = r_2(y_2) = s_2$, showing that $S_1 \times S_2 \subset \psi_{y_1,y_2}(P_{<d})$, as desired.

We now construct such R_1 and R_2. Let R_1 and R_2 be arbitrary size $K/2$ subsets of $\sigma(P_{<n-(s+1)})$. Note that since σ is injective, $\left|\sigma(P_{<n-(s+1)})\right| = q^{n-s-1} \geq K/2$ and so we can indeed pick such R_1 and R_2. We claim that for any $r_1 \in R_1, r_2 \in R_2$, it holds that $(r_1, r_2) \in \varphi(P_{<n})$. Using Lemma 14, it suffices to show that $(r_1, r_2) = (f, f+\sigma(h))$ for some $f \in P_{<n}$ and $h \in P_{n-(s+1)}$. Since $R_1, R_2 \subseteq \sigma(P_{<n-(s+1)})$, we know that $r_1 = \sigma(h_1)$ and $r_2 = \sigma(h_2)$ for some $h_1, h_2 \in P_{<n-(s+1)}$. Consequently,

$$(r_1, r_2) = (\sigma(h_1), \sigma(h_2)) = (\sigma(h_1), \sigma(h_1) + \sigma(h_2 - h_1))$$

as desired. In the last line, we used the fact that σ is a homomorphism and that $h_1, h_2 \in P_{<n-(s+1)}$.

We finally define σ:

▶ **Definition 15.** *Recall that $\sigma : P_{<n-(s+1)} \to P_{\leq s}$. To compute $\sigma(h_1)$, we let $f \in P_{<n}$ be any polynomial such that $f = h_1 g_1 + r_1$ where $r_1 \in P_{\leq s}$ is the remainder of f modulo g_1. We then write $f = h_2 g_2 + r_2$ where $r_2 \in P_{\leq s}$ is the remainder of f modulo g_2. The output of $\sigma(h_1)$ is $r_2 - r_1$.*

At first glance, it seems unclear whether σ is even a well defined function. To help show this, we define $\rho : P_{<n-(s+1)} \to P_{<n-(s+1)}$ such that $\rho(h_1) = h_2$ where h_1, h_2 are defined as above. We first show that ρ is an isomorphism. To do this, we observe that

$h_1g_1 - h_2g_2 = r_2 - r_1$ has degree at most s, and so it must be that h_1g_1 and h_2g_2 agree on all coefficients corresponding to degree $\geq s+1$. We compare these coefficients and, using linear algebra, show that we can indeed obtain a unique h_2 from h_1, showing it is indeed a function. This argument in fact directly shows that ρ is an isomorphism.

Once we have that ρ is an isomorphism, we can then define $\sigma(h_1) = h_1g_1 - \rho(h_2)g_2$. Using this, it easily follows that σ is a homomorphism. From this we obtain Lemma 14. By a further counting argument, we can show that σ is injective. For details, see Section 5.2.

2.2 Organization

We use Section 3 to introduce necessary preliminaries. Then in Section 4.1 we show how our main theorem is proved assuming right regularity and knowing the overlap between two neighborhoods of right vertices. These facts are then proved in Section 4.2. In Section 4.3, we plug in parameters to get our two-sided lossless expander. In Section 5 we prove tightness of our right-to-left expansion analysis of the KT graph.

We prove that our constructions are explicit in Appendix A, and discuss why our techniques do not work for the [10] graph in Appendix B.

3 Preliminaries

3.1 Notation

For a function $f \in \mathbb{F}_q[X]$, we we use $f^{(j)}$ to denote the j'th iterated derivative of f. We will often use the notation b_i for $i \in \mathbb{N}$ to refer to the polynomial $x^i \in \mathbb{F}_q[x]$ and we will often use the fact that $(b_0, \ldots, b_n)$ form a basis for the polynomials of degree at most n. For a (D_L, D_R)-biregular bipartite graph $G = (L \sqcup R)$, we use $\Gamma_\rightarrow : L \times [D_L] \to R$ to be the function that maps vertices in L to their neighbors in R as given by G; we use $\Gamma_\leftarrow : R \times [D_R] \to L$ to be the function that maps vertices in R to their neighbors in L as given by G. Often, we will define graph G by only defining the associated $\Gamma_\rightarrow$. When clear from context, we sometimes abuse notation and use $\Gamma_\rightarrow(w)$ to denote the right neighborhood of $w \in L$, and similarly $\Gamma_\leftarrow(w)$ for the left neighborhood of $w \in R$.

3.2 Lossless expansion

Throughout this paper, we will be focusing on the notion of vertex expansion as opposed to other definitions (e.g., edge, spectral) of expansion. Defining vertex expansion of a regular graph is straightforward.

▶ **Definition 16.** *A D-regular graph $G = (V, E)$ is a* (K, A)-expander *if for all $S \subseteq V$ such that $|S| \leq K$ we have that $|\Gamma(S)| \geq A\,|S|$. If $A = 1 - \varepsilon$, then we say that G is a* (K, ε)-lossless expander.

For biregular bipartite graphs, we must consider the degree of each side to define expansion.

▶ **Definition 17.** *A (D_L, D_R)-biregular graph $G = (L \sqcup R, E)$ is a* (K_L, A_L, K_R, A_R)-two-sided expander *if for all $S \subseteq L$ of size at most K_L we have $|\Gamma_\rightarrow(S)| \geq A_L\,|S|$ and for all $S \subseteq R$ of size at most K_R we have $|\Gamma_\leftarrow(S)| \geq A_R\,|S|$. If $A_L = 1 - \varepsilon_L$ and $A_R = 1 - \varepsilon_R$, then we call G a* $(K_L, \varepsilon_L, K_R, \varepsilon_R)$-lossless two-sided expander.

For irregular graphs, we can generalize Definition 16 in two ways. The first way is considering expansion with respect to the maximum number of neighbors of a set.

▶ **Definition 18.** *An irregular graph $G = (V, E)$ is a* (K, ε)-lossless expander *(where we abuse the word "expander" for both regular and irregular graphs) if for any set $S \subseteq V$ of size at most K we have that $|\Gamma(S)| \geq (1 - \varepsilon) \sum_{v \in S} d(v)$ where $d(v)$ represents the degree of vertex v.*

The second, stronger notion of lossless expansion is with respect to the highest degree of a node present in a graph.

▶ **Definition 19.** *An irregular graph $G = (V, E)$ is a* max-degree (K, ε)-lossless expander *if for any set $S \subseteq V$ of size at most K we have that $|\Gamma(S)| \geq (1 - \varepsilon) D |S|$ where $D = \max_{v \in V} d(v)$, the maximum degree of any vertex in G.*

3.3 The KT graph

Throughout the paper, we will use construction of bipartite (left) lossless expanders from [15] based on multiplicity codes from [16]. We will often refer to this graph "the KT graph":

▶ **Definition 20** (The KT graph)**.** *Let $q, n, s \in \mathbb{N}$ be such that q is a prime power, characteristic of the finite field $\mathbb{F}_q \geq n$ and $s \leq n/2$. Define $G = (L \sqcup R, E)$ where $L = \mathbb{F}_q^n, R = \mathbb{F}_q^{s+2}$. The left degree is q and for any $f \in \mathbb{F}_q^n$ and $y \in \mathbb{F}_q$, the y'th neighbor of f is defined as follows: Identify f as member of $\mathbb{F}_q[X]$ with degree of f at most $n - 1$; then, the neighbor $\Gamma_{\rightarrow}(f, y)$ will be $(y, f^{(0)}(y), \ldots, f^{(s)}(y))$ where $f^{(j)}$ is the j'th iterative derivative of f.*

▶ Remark 21. In the paper [15], the final lossless expander graph construction slightly differs from ours. While they do construct the KT-graph G defined as above and show it has great (left) expanding properties, the final (left) lossless expander graph actually is defined as $H = (L \sqcup R, E)$ where $L = 2^n, R = \mathbb{F}_q^{s+2}$ and the left degree is q. H is constructed by considering the subgraph of G induced by vertices on the left side corresponding to $\{0, 1\}^n$. For us, the final two-sided lossless expander graph will be G itself. This is why, our two-sided lossless expander graph has slightly worse parameters (worse constants) compared to the left lossless expander graph from [15].

3.4 A useful inequality

We will use the following inequality based on an application of the Cauchy-Schwarz inequality:

▷ Claim 22. Fix $n \in \mathbb{N}, S \in \mathbb{R}$. Let $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$ be such that $\sum_{1 \leq i \leq n} x_i = S$. Then,

$$\sum_{1 \leq i < j \leq n} x_i x_j \leq \frac{(n-1)S^2}{2n}$$

Proof. Recall the Cauchy-Schwarz inequality:

$$\left(\sum_{1 \leq i \leq n} a_i b_i \right)^2 \leq \left(\sum_{1 \leq i \leq n} a_i^2 \right) \left(\sum_{1 \leq i \leq n} b_i^2 \right).$$

We apply this with $a_1 = x_1, \ldots, a_n = x_n$ and $b_1 = b_2 = \cdots = b_n = 1$ to infer that

$$S^2 \leq \left(\sum_{1 \leq i \leq n} x_i^2 \right) \cdot n = \left(S^2 - 2 \sum_{1 \leq i < j \leq n} x_i x_j \right) \cdot n$$

Rearranging, we infer that

$$\sum_{1\le i<j\le n} x_i x_j \le \frac{(n-1)S^2}{2n}$$

as desired. ◁

4 An Explicit Two-sided Lossless Expander

In this section, we first describe how to prove our main theorem using right regularity and the size of the overlap in neighborhoods between any two right vertices. Then we prove these two facts in Section 4.2.

4.1 Main theorem

Putting together all of our results with the left-to-right expansion of [15] yields our main theorem.

▶ **Theorem 23.** *For all finite fields $\mathbb{F}_q$ and $n, s \in \mathbb{N}$ with $15 \le (s+1) < n < char(\mathbb{F}_q)$, there exists an explicit bipartite graph $G = (L \sqcup R, E)$ with $L = \mathbb{F}_q^n, R = \mathbb{F}_q^{s+2}$, left degree equal to q and right degree $q^{n-(s+1)}$ such that G is a two-sided (K_L, A_L, K_R, A_R) expander with $A_L = q - \frac{n(s+2)}{2} \cdot (qK_L)^{1/(s+2)}$ and $A_R = \left(1 - \frac{K_R}{q^{\min(s+2,n-s)}} \cdot \frac{q-1}{2}\right) q^{n-(s+1)}$.*

Proof. The left-to-right expansion follows from Theorem 3 from [15]. The right-to-left expansion follows from Theorem 24 below. The explicitness of G follows from Claim 47. ◀

Our main achievement is showing the right-to-left expansion of the KT graph in Theorem 24 below.

▶ **Theorem 24.** *If $n \ge s+1$, then the KT graph G in Definition 6 is a right (K_{max}, ε)-lossless expander for $K_{max} = \delta q^{s+1}$ and $\varepsilon = \frac{\delta(q-1)}{2q} \cdot q^{\max(2s+2-n,0)}$ where $0 < \delta < 1$ is arbitrary.*

We prove Theorem 24 via the following properties of G:

▶ **Lemma 25.** *When $n \ge s+1$, G is right-regular and the right degree is $q^{n-(s+1)}$.*

▶ **Lemma 26.** *For any pair of right-vertices w_1, w_2 such that $w_1 = (y_1, z_1), w_2 = (y_2, z_2) \in \mathbb{F}_q^{s+2}$ where $y_1 \ne y_2 \in \mathbb{F}_q$ and $z_1, z_2 \in \mathbb{F}_q^{s+1}$, we have*

$$|\Gamma_{\leftarrow}(y_1, z_1) \cap \Gamma_{\leftarrow}(y_2, z_2)| \le \begin{cases} q^{n-(2s+2)} & n \ge 2s+2 \\ 1 & n \le 2s+2 \end{cases}$$

We will prove both these lemmas in Section 4.2.

With the exact right-regularity of G and the number of common left-neighbors shared by any pair of right-vertices generated by different seeds, we are ready to prove Theorem 24.

Proof of Theorem 24. Our goal is to show that any right subset $T \subseteq \mathbb{F}_q^{s+2}$ of size at most δq^{s+1} has a neighborhood of size at least $(1-\varepsilon)q^{n-(s+1)}|T|$ on the left.

To do this, we consider T as the disjoint union $T = \bigsqcup_{y \in \mathbb{F}_q} T_y$ of buckets $T_y = \{(y, \alpha) : \alpha \in \mathbb{F}_q^{s+1}\}$ where $|T_y| = t_y = \delta_y q^{s+1}$. Let $\delta = \sum_{y \in \mathbb{F}_q} \delta_y$. So, $|T| = \delta q^{s+1}$. By Lemma 25, the number of edges leaving T is $|T| \cdot q^{n-(s+1)} = \delta q^n$.

We now consider cases on whether $n \ge 2s+2$ or not:

Case 1. $n \geq 2s+2$.

In this case, $\varepsilon = \frac{\delta(q-1)}{2q}$. By Lemma 26, the maximum number of double-counted left vertices is

$$\sum_{\substack{i,j\in[q]\\ i<j}} t_i t_j q^{n-2(s+1)} = \sum_{\substack{i,j\in[q]\\ i<j}} \delta_i q^{s+1} \cdot \delta_j q^{s+1} \cdot q^{n-2(s+1)} = q^n \sum_{\substack{i,j\in[q]\\ i<j}} \delta_i \delta_j \leq q^n \cdot \frac{q-1}{2q} \cdot \delta^2$$

where for the last inequality, we used Claim 22. Applying one level of inclusion-exclusion reveals that

$$|\Gamma_{\leftarrow}(T)| \geq \delta q^n - q^n \cdot \frac{q-1}{2q} \cdot \delta^2 = \left(1 - \frac{\delta(q-1)}{2q}\right) \delta q^n = (1-\varepsilon)\, q^{n-(s+1)} |T|$$

where the last equality follows because $\varepsilon = \frac{\delta(q-1)}{2q}$.

Case 2. $2s+2 \geq n \geq s+1$.

In this case, $\varepsilon = \frac{\delta(q-1)}{2q} \cdot q^{2s+2-n}$. By Lemma 26, the maximum number of double-counted left vertices is

$$\sum_{\substack{i,j\in[q]\\ i<j}} t_i t_j = \sum_{\substack{i,j\in[q]\\ i<j}} \delta_i q^{s+1} \cdot \delta_j q^{s+1} = q^{2s+2} \sum_{\substack{i,j\in[q]\\ i<j}} \delta_i \delta_j \leq q^{2s+2} \cdot \frac{q-1}{2q} \cdot \delta^2$$

where for the last inequality, we used Claim 22. We again apply one level of inclusion-exclusion to conclude that

$$|\Gamma_{\leftarrow}(T)| \geq \delta q^n - q^{2s+2} \cdot \frac{q-1}{2q} \cdot \delta^2 = \left(1 - \frac{\delta(q-1)}{2q} \cdot q^{2s+2-n}\right) \delta q^n = (1-\varepsilon)\, q^{n-(s+1)} |T|$$

where the last equality follows because $\varepsilon = \frac{\delta(q-1)}{2q} \cdot q^{2s+2-n}$. ◀

4.2 Right regularity and bounding common neighbors: Hermite interpolation

In this section, we show the $(q^{n-(s+1)})$-right-regularity of the KT graph G, and bound the number of common left neighbors shared by any pair of right vertices with different seeds. Both tasks are essentially a question of Hermite interpolation – we wish to find polynomials $f \in \mathbb{F}_q[Y]$ of degree at most $n-1$ such that when evaluating at some point $y \in \mathbb{F}_q$, the function value $f(y)$ and its first s derivatives $(f^{(0)}(y), f^{(1)}(y), \cdots, f^{(s)}(y))$ match the values given by the right vertices.

▶ **Lemma 27** (Hermite interpolation). *Let $y_1, \cdots, y_k \in \mathbb{F}_q$ be distinct, and for $i \in [k]$, let $z_{i,0}, \cdots, z_{i,s} \in \mathbb{F}_q$. Then there exists a unique polynomial $f \in \mathbb{F}_q[Y]$ with degree at most $k(s+1)$ such that $f^{(j)}(y_i) = z_{i,j}$ for $i \in [k]$ and $j \in \{0\} \cup [s]$.*

Proof. Consider the following k congruences,

$$f(Y) \equiv f_1(Y) \mod (Y-y_1)^{s+1}, \ \cdots, \ f(Y) \equiv f_k(Y) \mod (Y-y_k)^{s+1}$$

Any polynomial f that satisfies the above k congruences must also satisfy $f^{(j)}(y_i) = z_{i,j}$ for $i \in [k]$ and $j \in \{0\} \cup [s]$ – thus solving the interpolation – because $f_i(Y)$ is the order s Taylor polynomial of f at y_i.

We conclude the proof by applying the Chinese remainder theorem for univariate polynomials which asserts that there exists a *unique* polynomial $f \in \mathbb{F}_q[Y]$ of degree at most $k(s+1)$ that satisfies the above congruences. ◀

Recall the maps $\psi_{y_1} : \mathbb{F}_q^n \to \mathbb{F}_q^{s+1}$, and $\psi_{y_1,y_2} : \mathbb{F}_q^n \to \mathbb{F}_q^{2s+2}$ where we have $\psi_y(f) = (f^{(0)}(y), \ldots, f^{(s)}(y))$ and $\psi_{y_1,y_2} = (\psi_{y_1}(f), \psi_{y_2}(f))$. We will use the following fact regarding linearity of derivatives:

▶ **Fact 28** ([19]). *For all $\alpha, \beta \in \mathbb{F}_q$, $f, g \in \mathbb{F}_q[X]$ and $j \geq 0$, it holds that $(\alpha f + \beta g)^{(j)} = \alpha f^{(j)} + \beta g^{(j)}$.*

From this fact, we directly obtain that ψ_y is a linear map:

▶ **Corollary 29.** *For all $y \in \mathbb{F}_q$, ψ_y is an $\mathbb{F}_q$-linear map.*

Now we prove the right-regularity of G (Lemma 25) and bound the number of overlapping left neighbors (Lemma 26) as special cases of Lemma 27.

Proof of Lemma 25. Let $w_1 \in \mathbb{F}_q^{s+2}$ where $w_1 = (y_1, z_{1,0}, \cdots, z_{1,s})$. Take $k = 1$ in Lemma 27, we get that there exists a unique polynomial f of degree at most $s+1$ such that $f^{(j)}(y_1) = z_{1,j}$ for $j \in \{0\} \cup [s]$. This means the linear map $\psi_{y_1} : \mathbb{F}_q^n \to \mathbb{F}_q^{s+1}$ is surjective. Therefore, the number of left neighbors of any right vertex w_1 is exactly $|\psi_{y_1}^{-1}(z_{1,0}, \cdots, z_{1,s})| = q^{n-(s+1)}$. ◀

Proof of Lemma 26. Let $w_1, w_2 \in \mathbb{F}_q^{s+2}$ where $w_1 = (y_1, z_{1,0}, \cdots, z_{1,s})$ and similarly $w_2 = (y_2, z_{2,0}, \cdots, z_{2,s})$ with $y_1 \neq y_2$. Take $k = 2$ in Lemma 27, we get that there exists a unique polynomial f of degree at most $2(s+1)$ such that $f^{(j)}(y_i) = z_{i,j}$ for $i \in \{1,2\}, j \in \{0\} \cup [s]$. This means,

- When $n > 2s+2$, $\psi_{y_1,y_2} : \mathbb{F}_q^n \to \mathbb{F}_q^{2s+2}$ is surjective, the number of common neighbors shared by w_1, w_2 is exactly $|\psi_{y_1,y_2}^{-1}(z_{1,0}, \cdots, z_{1,s}, z_{2,0}, \cdots, z_{2,s})| = q^{n-2(s+1)}$.
- When $n \leq 2s+2$, $\psi_{y_1,y_2} : \mathbb{F}_q^n \to \mathbb{F}_q^{2s+2}$ is injective, w_1, w_2 share *at most* 1 common left neighbor. ◀

4.3 Plugging in the Parameters

We record our main results regarding two sided lossless expanders:

▶ **Theorem 30** (Formal version of Theorem 2). *For infinitely many N and all $0 < \delta < 0.99$, there exists an explicit biregular two-sided $(K_L, \varepsilon_L = 0.01, K_R, \varepsilon_R = 0.01)$ lossless expander $\Gamma_\to : [N] \times [D_L] \to [M]$ where $D_L \leq O(\log^{204}(N))$, $N^{1.01\cdot\delta - o(1)} \leq M \leq D_L \cdot N^{1.01\cdot\delta}$, $K_L = N^\delta$, $K_R = \frac{1}{50} \cdot (1/D_L) \cdot \min(M, N/M)$.* [7]

These will follow from the following technical lemma:

▶ **Lemma 31.** *Let $\alpha, \varepsilon_L, \varepsilon_R \in (0,1)$ and $K_R, n, k_L, q \in \mathbb{N}$ be such that q is a prime number, $\frac{h^{1+\alpha}}{2} \leq q \leq h^{1+\alpha}$ where $h = (4nk_L/\varepsilon_L)^{1/\alpha}$ and such that both $\frac{4}{k_L}\log(2n/\varepsilon_L) \leq \alpha$ and $k_L(1+\alpha) \leq n$. Then, there exists an explicit biregular $(K_L, \varepsilon_L, K_R, \varepsilon_R)$ two-sided lossless expander $\Gamma_\to : [N] \times [D_L] \to [M]$ where $N = q^n, K_L = q^{k_L}, K_L^{1+\alpha-1/\log(h)} \leq M \leq D_L \cdot K_L^{1+\alpha}$, $D_L \leq O(\log(N)\log(K_L)/\varepsilon_L)^{1+1/\alpha+o(1)}$, $\frac{K_R}{q^{\min(s+2,n-s)}} \cdot \frac{q-1}{2} \leq \varepsilon_R$ where $s+2 = \lceil k_L/\log_q(h) \rceil$.*

We will instantiate this lemma using simple parameters to obtain our main theorems:

Proof of Theorem 30. We plug in $\alpha = 0.01, \varepsilon_L = 0.01, \varepsilon_R = 0.01, k_L = \delta n$ in Lemma 31 to obtain the desired lossless expander. ◀

[7] Our theorem statement doesn't have any additional constraint on D_R since it can be uniquely inferred from N, M, D_L.

We finally prove our main technical lemma using two-sided expander from Theorem 23:

Proof of Lemma 31. As $s+2 = \lceil k_L/\log_q(h) \rceil$, we have that $h^{s+1} \leq K_L \leq h^{s+2}$. Observe that

$$s+1 < \frac{k_L \log(q)}{\log(h)} \leq k_L(1+\alpha) \leq n$$

So, we can apply Theorem 23 and infer that there exists a graph $\Gamma_{\rightarrow} : \mathbb{F}_q^n \times \mathbb{F}_q \rightarrow \mathbb{F}_q^{s+2}$ that is a $(\leq h^{s+2}, A_L)$ left expander and $(\leq K_R, A_R)$ right expander where $A_L = q - \frac{n(s+2)}{2} \cdot (qh^{s+2})^{1/(s+2)}$ and $A_R = \left(1 - \frac{K_R}{q^{\min(s+2,n-s)}} \cdot \frac{q-1}{2}\right) q^{n-(s+1)}$. Notice that as $K_L \leq h^{s+2}$, $\Gamma_{\rightarrow}$ is indeed a (K_L, A_L) expander.

- We first bound the left degree D_L:

  $$\begin{aligned} D_L = q &\leq h^{1+\alpha} = (4nk_L/\varepsilon_L)^{1+1/\alpha} = (4\log(N)\log(K_L)/\log^2(q)\varepsilon_L)^{1+1/\alpha} \\ &= (4\log(N)\log(K_L)/\varepsilon_L)^{1+1/\alpha} \cdot \log^{2+2/\alpha}(q) \end{aligned}$$

  This implies that

  $$D_L = q \leq (4\log(N)\log(K_L)/\varepsilon_L)^{1+1/\alpha} \left(\log\left(8\log(N)\log(K_L)/\varepsilon_L\right)\right)^{2+2/\alpha}$$

  Then indeed, $D_L \leq O(\log(N)\log(K_L)/\varepsilon_L)^{1+1/\alpha+o(1)}$.
- We now bound the number of right vertices M:

  $$M = q^{s+2} \leq q \cdot h^{(1+\alpha)(s+1)} \leq q \cdot K_L^{1+\alpha}$$

  Additionally,

  $$M = q^{s+2} \geq q^{K_L \log(q)/\log(h)} \geq q^{K_L((1+\alpha)(\log h)-1)/\log(h)} = q^{K_L(1+\alpha)-K_L/\log(h)}$$

- We now show lossless expansion from the right side:

  $$A_R = \left(1 - \frac{K_R}{q^{s+2}} \cdot \frac{q-1}{2}\right) q^{n-(s+1)} \geq (1-\varepsilon_R) D_R$$

  where the last inequality follows because $\frac{K_R}{q^{\min(s+2,n-s)}} \cdot \frac{q-1}{2} \leq \varepsilon_R$.
- We finally show lossless expansion from the left side: First, we note that $s+2 \leq 2k_L$. Indeed, $s+2 \leq k_L \log_q(h) + 1 = k_L \frac{\log(h)}{\log(q)} + 1 \leq k_L(1+\alpha)+1 \leq 2k_L$. Then,

  $$\begin{aligned} A_L &= q - \frac{n(s+2)}{2} \cdot (qh^{s+2})^{1/(s+2)} \\ &= q - \frac{n(s+2)h}{2} \cdot (q)^{1/(s+2)} \\ &\geq q - nk_L h \cdot (q)^{1/(s+2)} && \text{(since } s+2 \leq 2k_L\text{)} \\ &= q - \frac{\varepsilon_L \cdot h^{\alpha}}{4} \cdot h \cdot (q)^{1/(s+2)} && \text{(since } nk_L = (\varepsilon_L \cdot h^{\alpha})/4\text{)} \\ &= q - \varepsilon_L \cdot \frac{h^{1+\alpha}}{4} \cdot (q)^{1/(s+2)} \\ &\geq q - \frac{\varepsilon_L}{2} \cdot q \cdot (q)^{1/(s+2)} && \text{(since } h^{1+\alpha}/2 \leq q\text{)} \\ &= q\left(1 - \varepsilon_L \cdot \frac{(q)^{1/(s+2)}}{2}\right) \\ &\geq q(1-\varepsilon_L) \end{aligned}$$

The last inequality $(q)^{1/(s+2)} \leq 2$ follows because we claim that $s+2 \geq \log(q)$. This suffices to prove the last inequality since then $(q)^{1/(s+2)} \leq q^{1/\log(q)} \leq 2$. We indeed compute that

$$s+2 \geq \frac{k_L}{\log_q(h)} \geq \frac{k_L((1+\alpha)\log(h)-1))}{\log(h)} \geq k_L$$

Moreover, as $\alpha \geq \frac{4}{k_L}\log(2n/\varepsilon_L)$, we infer that $k_L \geq \frac{4}{\alpha}\cdot\log(2n/\varepsilon_L)$. Hence indeed,

$$s+2 \geq \frac{4}{\alpha}\cdot\log(2n/\varepsilon_L) \geq \frac{2}{\alpha}\cdot\log(2nk_L/\varepsilon_L) \geq 2\log(h) \geq (1+\alpha)\log(h) \geq \log(q) \quad \blacktriangleleft$$

5 Tightness of Our Construction

In this section we show that the right-to-left expansion of Theorem 24 is tight. In particular, when $n < 2s+2$ Theorem 24 gives a trade-off between the expansion parameter ε and the max size of expanding sets K_{max}. We show that this trade-off is tight up to constants, and thus fully characterize the behavior of the KT graph. Importantly, we show that in the balanced setting where $n = s + O(1)$, the KT graph is *not* a two-sided lossless expander.

Recall that when $n \geq 2s+2$ we know that our result is tight as stated in Remark 3. Consequently, our main theorem in this section deals with the regime where $s+1 < n < 2s+2$. In this setting, Theorem 24 gives us that sets $S \subseteq R$ on the right of size at most $K_{max} = \delta q^{s+1}$ expand with parameter $\varepsilon = \frac{\delta(q-1)}{2q}\cdot q^{2s+2-n}$. Equivalently, it gives us that sets of size at most $K_{max} = \delta q^{n-s-1}$ expand with parameter $\varepsilon = \frac{\delta(q-1)}{2q}$. Our main theorem in this section upper bounds this expansion.

▶ **Theorem 32.** *When $s+1 < n < 2s+2$, there exists a subset $S \subseteq R$ of the right vertices such that $|S| = K_{max} = \delta q^{n-s-1}$ and $|\Gamma_{\leftarrow}(S)| = (1-\varepsilon)D_R|S|$ with $\varepsilon = \frac{\delta}{4}$ where $\delta > 0$.*

This means that our right-to-left expansion of *Theorem* 24 is tight up to a constant factor of 1/2. The proof of our main theorem comes from the construction of two disjoint subsets of right vertices each in different buckets that have the maximum number of overlapping neighbors on the left. Recall that the y-th bucket T_y is defined as $T_y = \{(y,\alpha) : \alpha \in \mathbb{F}_q^{s+1}\}$

▶ **Lemma 33.** *Let $y_1, y_2 \in \mathbb{F}_q$ be arbitrary such that $y_1 \neq y_2$. Then there exist sets $S_1 \subseteq T_{y_1}$ and $S_2 \subseteq T_{y_2}$ such that $|S_1| = |S_2| = \frac{K_{max}}{2}$ and $|\Gamma_{\leftarrow}(S_1) \cap \Gamma_{\leftarrow}(S_2)| = |S_1|\cdot|S_2|$.*

Using this lemma, our main theorem is a result of straightforward computations.

Proof of Theorem 32. Take any $y_1, y_2 \in \mathbb{F}_q$ such that $y_1 \neq y_2$ and the S_1 and S_2 from Lemma 33. Let $S = S_1 \cup S_2$, so indeed $|S| = K_{max}$. To count $|\Gamma_{\leftarrow}(S)|$, all we need to do is to subtract the number of 2-wise overlaps of left neighbors S_1 and S_2 from the total number of possible neighbors of S_1 and S_2. The latter value is simply $|S|\cdot D_R = K_{max}\cdot D_R$, and the former is given to us by Lemma 33 as $|S_1|\cdot|S_2| = \frac{K_{max}^2}{4}$. Therefore, we can compute

$$|\Gamma_{\leftarrow}(S)| = K_{max}\cdot D_R - \frac{K_{max}^2}{4} = K_{max}\cdot D_R\left(1 - \frac{K_{max}}{4D_R}\right),$$

showing that

$$\varepsilon = \frac{K_{max}}{4D_R} = \frac{\delta q^{n-(s+1)}}{4q^{n-(s+1)}} = \frac{\delta}{4}. \quad \blacktriangleleft$$

The proof of Lemma 33 relies on choosing S_1 and S_2 to contain only the derivatives of polynomials of degree at most $n-(s+1)$. In the following proofs, we will consider vector spaces of polynomials over $\mathbb{F}_q$.

▶ **Definition 34.** *Define the set* $P_d = \{f \in \mathbb{F}_q[x] \mid \deg(f) = d\}$ *and let* $P_{<d} = \{f \in \mathbb{F}_q[x] \mid \deg(f) < d\}$ *be a vector space over* $\mathbb{F}_q$ *with addition and multiplication coming from* $\mathbb{F}_q[x]$.

5.1 Constructing sets with the smallest expansion possible

We prove Lemma 33 by restricting the KT graph to the subgraph containing the two buckets T_{y_1} and T_{y_2} and analyzing the behavior of this subgraph. Consequently, we rephrase Lemma 33 as follows.

▶ **Lemma 35** (Technical version of Lemma 33). *For any* $K \in [q^{n-(s+1)}]$ *and distinct* $y_1, y_2 \in \mathbb{F}_q$, *there exist sets* $S_1, S_2 \subseteq \mathbb{F}_q^{s+1}$ *such that* $|S_1| + |S_2| = K$ *and* $S_1 \times S_2 \in \psi_{y_1,y_2}(P_{<n})$.

To create these S_1 and S_2, we actually first create analogous sets in the image of the Chinese Remainder Theorem map, defined below.

▶ **Definition 36.** *For distinct* $y_1, y_2 \in \mathbb{F}_q$ *let* $g_1(x) = (x-y_1)^{s+1}$ *and* $g_2(x) = (x-y_2)^{s+1}$, *let* $\pi_1 : \mathbb{F}_q[x] \to \mathbb{F}_q[x]/(g_1)$ *and* $\pi_2 : \mathbb{F}_q[x] \to \mathbb{F}_q[x]/(g_2)$ *be the associated quotient maps. Then let* $\varphi : \mathbb{F}_q[x]/(g_1g_2) \to \mathbb{F}_q[x]/(g_1) \times \mathbb{F}_q[x]/(g_2)$ *be defined as* $\varphi = \pi_1 \otimes \pi_2$.

This is exactly the map that the Chinese Remainder Theorem acts on.

▷ Claim 37. The Chinese Remainder Theorem says that φ is an isomorphism of rings.[8]

▶ **Corollary 38.** *As a consequence, we may also think of* φ *as an isomorphism of* $\mathbb{F}_q$*-vector spaces* $\varphi : P_{<2s+2} \to P_{<s+1} \times P_{<s+1}$.

Whereas $\psi_{y_1,y_2}(f)$ tells us about the first s derivatives of f at y_1 and y_2, $\varphi(f)$ tells us about f quotiented by g_1 and g_2. In fact, as we will see, φ and ψ_{y_1,y_2} provide us with the same information about a particular polynomial. With this in mind, we prove a lemma similar to Lemma 35 but for φ.

▶ **Lemma 39.** *For any* $K \in [q^{n-(s+1)}]$ *and distinct* $y_1, y_2 \in \mathbb{F}_q$, *there exist sets* $S_1 \subseteq \mathbb{F}_q[x]/(g_1)$ *and* $S_2 \subseteq \mathbb{F}_q[x]/(g_2)$ *such that* $|S_1| + |S_2| = K$ *and* $S_1 \times S_2 \in \varphi(P_{<n})$.

Using this relation between φ and ψ_{y_1,y_2}, we prove Lemma 35.

Proof of Lemma 35. Use Lemma 39 to create $R_1 \subseteq \mathbb{F}_q[x]/(g_1)$ and $R_2 \subseteq \mathbb{F}_q[x]/(g_2)$ such that $|R_1| + |R_2| = K$ and $R_1 \times R_2 \in \varphi(P_{<n})$. Let $S_1 = \psi_{y_1} \circ \pi_1^{-1}(R_1)$ and $S_2 = \psi_{y_2} \circ \pi_2^{-1}(R_2)$ where π_1^{-1} and π_2^{-1} are the natural inclusions of $\mathbb{F}_q[x]/(g_1)$ and $\mathbb{F}_q[x]/(g_2)$ into $\mathbb{F}_q[x]$, respectively. We claim that $S_1 \times S_2 \in \psi_{y_1,y_2}(P_{<n})$.

Since the polynomials in the images of π_1^{-1} and π_2^{-1} are of degree strictly less than $s+1$, we have that $\psi_{y_1} \circ \pi_1^{-1}$ and $\psi_{y_2} \circ \pi_2^{-1}$ are injective, so $|S_1| + |S_2| = |R_1| + |R_2| = K$. Moreover, for a pair $(s_1, s_2) \in S_1 \times S_2$, we can take the unique $(r_1, r_2) \in R_1 \times R_2$ such that $s_1 = \psi_{y_1} \circ \pi_1^{-1}(r_1)$ and $s_2 = \psi_{y_2} \circ \pi_2^{-1}(r_2)$. Thus, $(s_1, s_2) = \psi_{y_1,y_2}(\pi_1^{-1}(r_1), \pi_2^{-1}(r_2)) \in \psi_{y_1,y_2}(P_{<n})$, as claimed. ◀

Our proof of Lemma 39 relies on a result about the structure of the image of φ which we prove in Section 5.2 but state here.

[8] For an introduction to the Chinese Remainder Theorem, see Chapter 7.6 of [6].

▶ **Definition 40.** *For any $b \in \mathbb{F}_q^{s+1}$ define the line $\ell_b = \{(f, f+b) \mid f \in \mathbb{F}_q^{s+1}\}$.*

We now show that the image of φ is actually composed of many of these lines with shifts given by the injective homomorphism $\sigma : P_{<n-(s+1)} \to P_{<s+1}$ that we introduce later in Definition 45.

▶ **Lemma 41.** *We show that:*
1. $\varphi(P_{<s+1}) = \ell_0$
2. *For $2s+2 > d \geq s+1$ we have $\varphi(P_d) = \bigcup_{h \in P_{d-(s+1)}} \ell_{\sigma(h)}$.*

Thus, we can conclude that $\varphi(P_{<n}) = \bigcup_{h \in P_{<n-(s+1)}} \ell_{\sigma(h)}$ where σ is an injective homomorphism.

This structural result on the image of φ allows us to prove Lemma 39.

Proof of Lemma 39. Let $S_1, S_2 \subseteq \sigma(P_{<n-(s+1)})$ such that $|S_1| = |S_2| = \frac{K_{max}}{2}$. Recall that σ is injective so $\left|\sigma(P_{<n-(s+1)})\right| = \left|P_{<n-(s+1)}\right| = q^{n-(s+1)}$ and we have enough elements to choose from. We claim that any such choice of S_1 and S_2 satisfies the lemma statement.

To prove this claim, we have to show that any pair $(s_1, s_2) \in S_1 \times S_2$ lies in $\varphi(P_{<n})$. By Lemma 41, this is equivalent to saying that (s_1, s_2) is of the form $(f, f + \sigma(h))$ for some $f \in P_{<s+1}$ and $h \in P_{<n-(s+1)}$. From our construction, we have that $s_1 = \sigma(h_1)$ and $s_2 = \sigma(h_2)$ for $h_1, h_2 \in P_{<n-(s+1)}$. So we are considering the point $(\sigma(h_1), \sigma(h_2))$. Using the fact that σ is a homomorphism from Claim 46, we can rewrite this point in our desired form:

$$\begin{aligned}(\sigma(h_1), \sigma(h_2)) &= (\sigma(h_1), \sigma(h_1) + \sigma(h_2) - \sigma(h_1)) \\ &= (\sigma(h_1), \sigma(h_1) + \sigma(h_2 - h_1)).\end{aligned}$$

Thus, $(s_1, s_2) \in \ell_{\sigma(h_2 - h_1)} \subseteq \varphi(P_{<n})$, as claimed. ◀

5.2 The structure of the image of φ

We end Section 5 by proving Lemma 41, that $\varphi(P_{<n}) = \bigcup_{h \in P_{<n-(s+1)}} \ell_{\sigma(h)}$. To do so we first define a new homomorphism ρ which we will use to define σ later on.

▶ **Definition 42.** *Let $\rho : P_{<n-(s+1)} \to P_{<n-(s+1)}$ be defined as follows. Given $h_1 \in P_{<n-(s+1)}$, let $f \in P_{<n}$ be such that $f = h_1 g_1 + r_1$ for some $r_1 \in P_{<s+1}$. Then let $h_2 \in P_{\deg(f)-(s+1)}$ and $r_2 \in P_{<s+1}$ be the unique polynomials such that $f = h_2 g_2 + r_2$. We define $\rho(h_1) := h_2$.*

Given this definition, the natural first question is whether ρ is even well-defined since there are q^{s+1} many choices of f that could be used. Lemma 43 shows that ρ is well-defined and that it is, in fact, just an invertible linear operator, meaning that ρ is an automorphism of $P_{<n-(s+1)}$.

▶ **Lemma 43.** *Let $f \in P_{<n}$ and take $h_1, h_2, r_1, r_2 \in \mathbb{F}_q[x]$ to be the unique polynomials such that*

$$f = h_1 g_1 + r_1$$
$$f = h_2 g_2 + r_2$$

where $\deg(h_1) = \deg(h_2) = \deg(f) - (s+1)$ and $\deg(r_1), \deg(r_2) < s+1$.

Then h_2 can be determined uniquely from h_1, g_1, and g_2. Similarly, h_1 can be determined uniquely from h_2, g_1, and g_2.

Proof of Lemma 43. Rearranging the equations in the lemma statement gives us that

$$h_1 g_1 - h_2 g_2 = r_2 - r_1.$$

Since the degrees of r_1 and r_2 are at most s, we have that $\deg(r_2 - r_1) < s+1$ as well, meaning that $\deg(h_1 g_1 - h_2 g_2) < s+1$. Consequently, the coefficients of x^k for $k \in \{s+1, \ldots, n-1\}$ of $h_1 g_1$ and $h_2 g_2$ must match, which uniquely determines h_2 from h_1.

Formally, if for $t \in \{1,2\}$ we write $h_t(x) = \sum_{k=0}^{n-1-(s+1)} \eta_k^{(t)} x^k$ for $\eta_k^{(t)} \in \mathbb{F}_q$ and expand out $g_t(x) = \sum_{k=0}^{s+1} \gamma_k^{(t)} x^k$ where $\gamma_k^{(t)} \in \mathbb{F}_q$ and $\gamma_{s+1}^{(t)} = 1$, then for each $k \in \{s+1, \ldots, n-1\}$ we have the linear equation

$$\sum_{j=k-(s+1)}^{n-1-(s+1)} \gamma_{k-j}^{(1)} \eta_j^{(1)} = \sum_{j=k-(s+1)}^{n-1-(s+1)} \gamma_{k-j}^{(2)} \eta_j^{(2)}.$$

This yields $n-s-1$ linear equations, which we can write as $M^{(1)}\eta^{(1)} = M^{(2)}\eta^{(2)}$ where for $t \in \{1,2\}$ we let $\eta^{(t)} = (\eta_0^{(t)}, \ldots, \eta_{n-s-2}^{(t)})^\intercal$ and $M^{(t)} \in \mathbb{F}_q^{(n-s-1)\times(n-s-1)}$ is the upper triangular matrix defined as

$$M_{i,j}^{(t)} = \begin{cases} \gamma_{s+1+i-j}^{(t)} & i \le j \\ 0 & i > j \end{cases}$$

Since g_t is monic, meaning that $\gamma_{s+1}^{(t)} = 1$, we clearly see that $\det(M^{(t)}) = 1$, so it is invertible and $\eta^{(2)} = \left(M^{(2)}\right)^{-1} M^{(1)} \eta^{(1)}$. Similarly, $\eta^{(1)} = \left(M^{(1)}\right)^{-1} M^{(2)} \eta^{(2)}$. ◀

▶ **Corollary 44.** *ρ is an automorphism of $P_{<n-(s+1)}$.*

Proof. Lemma 43 shows that we can equivalently define ρ as $\rho = \left(M^{(2)}\right)^{-1} M^{(1)}$, which is an invertible linear transformation. ◀

Now that we have defined ρ and shown that it is an isomorphism, we use it to build σ.

▶ **Definition 45.** *Let $\sigma : P_{<n-(s+1)} \to P_{<s+1}$ be defined as $\sigma(h) = hg_1 - \rho(h)g_2$. Note that while we consider multiplication here to occur in $\mathbb{F}_q[x]$, Lemma 43 tells us that this difference yields a polynomial in $P_{<s+1}$.*

Unsurprisingly, σ is a homomorphism of vector spaces since ρ is one.

▷ Claim 46. Since ρ is an isomorphism, σ is a homomorphism of vector spaces over $\mathbb{F}_q$.

Proof. The fact that σ is a homomorphism of vector spaces over $\mathbb{F}_q$ follows from ρ being a homomorphism and addition and multiplication commuting in $\mathbb{F}_q[x]$. ◁

Finally, we are ready to prove Lemma 41, that $\varphi(P_{<n}) = \bigcup_{h \in P_{<n-(s+1)}} \ell_{\sigma(h)}$.

Proof of Lemma 41. We can directly show the first item by the fact that that the remainder of dividing a polynomial by another of larger degree is the polynomial itself. That is, for any polynomial $f \in P_{<s+1}$, we have that $\pi_1(f) = \pi_2(f) = f$, so $\varphi(f) = (f, f)$. Consequently, going over all f of degree less than $s+1$ gives us $\varphi(P_{<s+1}) = \{(f,f) : f \in P_{<s+1}\} = \ell_0$.

Now, when considering P_d for $2s+2 > d \ge s+1$, quotienting by a degree $s+1$ polynomial does have a non-trivial effect. That is, for $f \in P_d$, there must exist unique $h_1, h_2, r_1, r_2 \in \mathbb{F}_q[x]$ such that

$$f = h_1 g_1 + r_1$$
$$f = h_2 g_2 + r_2$$

where $\deg(h_1) = \deg(h_2) = \deg(f) - (s+1)$ and $\deg(r_1), \deg(r_2) < s+1$. Recall that $g_1(x) = (x-y_1)^{s+1}$ and $g_2(x) = (x-y_2)^{s+1}$. Moreover, $\pi_1(f) = r_1$ and $\pi_2(f) = r_2$. By definition, we have that $h_2 = \rho(h_1)$, meaning that $r_2 = r_1 + h_1 g_1 - \rho(h_1) g_2 = r_1 + \sigma(h_1)$. Therefore, we have $\varphi(f) = (r_1, r_1 + \sigma(h_1))$.

With this in mind, we recall that because h_1 and r_1 uniquely identify f, we can iterate over all elements of P_d by iterating over all $h_1 \in P_{d-(s+1)}$ and $r_1 \in P_{<s+1}$. That is, $P_d = \{h_1 g_1 + r_1 : h_1 \in P_{d-(s+1)}, r_1 \in P_{<s+1}\}$. From this we can conclude that

$$\begin{aligned}
\varphi(P_d) &= \{\varphi(f) : f \in P_d\} \\
&= \{\varphi(h_1 g_1 + r_1) : h_1 \in P_{d-(s+1)}, r_1 \in P_{<s+1}\} \\
&= \{(r_1, r_1 + \sigma(h_1)) : h_1 \in P_{d-(s+1)}, r_1 \in P_{<s+1}\} \\
&= \bigcup_{h_1 \in P_{d-(s+1)}} \ell_{\sigma(h_1)},
\end{aligned}$$

as claimed.

Lastly, to see that σ is injective we recall that φ is an isomorphism and count cardinalities. For the left hand side of $\varphi(P_{<n}) = \bigcup_{h \in P_{<n-(s+1)}} \ell_{\sigma(h)}$, we have that $|\varphi(P_{<n})| = |P_{<n}| = q^n$. For the right hand side, we recall that $|\ell_b| = q^{s+1}$ for any b, so $\left|\bigcup_{h \in P_{<n-(s+1)}} \ell_{\sigma(h)}\right| = \left|\sigma(P_{<n-(s+1)})\right| \cdot q^{s+1}$. Therefore, $\left|\sigma(P_{<n-(s+1)})\right| = \frac{q^n}{q^{s+1}} = q^{n-(s+1)}$, showing that σ is injective since $\left|P_{<n-(s+1)}\right| = q^{n-(s+1)}$. ◀

References

1. H. Buhrman, P. B. Miltersen, J. Radhakrishnan, and S. Venkatesh. Are Bitvectors Optimal? *SIAM Journal on Computing*, 31(6):1723–1744, January 2002. doi:10.1137/S0097539702405292.
2. Michael Capalbo, Omer Reingold, Salil Vadhan, and Avi Wigderson. Randomness conductors and constant-degree lossless expanders. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, STOC '02, pages 659–668, New York, NY, USA, May 2002. Association for Computing Machinery. doi:10.1145/509907.510003.
3. Yeyuan Chen. Unique-neighbor expanders with better expansion for polynomial-sized sets. In Yossi Azar and Debmalya Panigrahi, editors, *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pages 3335–3362. SIAM, 2025. doi:10.1137/1.9781611978322.108.
4. Itay Cohen, Roy Roth, and Amnon Ta-Shma. HDX Condensers. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1649–1664, November 2023. doi:10.1109/FOCS57990.2023.00100.
5. Dean Doron and Roei Tell. Derandomization with Minimal Memory Footprint. In *DROPS-IDN/v2/document/10.4230/LIPIcs.CCC.2023.11*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.CCC.2023.11.
6. David S. Dummit and Richard M. Foote. *Abstract Algebra*. John Wiley & Sons, 3 edition, July 2003. Google-Books-ID: KJDBQgAACAAJ.
7. Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the Method of Multiplicities, with Applications to Kakeya Sets and Mergers. *SIAM Journal on Computing*, 42(6):2305–2328, January 2013. doi:10.1137/100783704.
8. Oded Goldreich. In a World of P=BPP. In Oded Goldreich, editor, *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation: In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*, pages 191–232. Springer, Berlin, Heidelberg, 2011. doi:10.1007/978-3-642-22670-0_20.

9 Louis Golowich. New Explicit Constant-Degree Lossless Expanders. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, Proceedings, pages 4963–4971. Society for Industrial and Applied Mathematics, January 2024. doi:10.1137/1.9781611977912.177.

10 Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from Parvaresh–Vardy codes. *Journal of the ACM*, 56(4):20:1–20:34, July 2009. doi:10.1145/1538902.1538904.

11 Jun-Ting Hsieh, Ting-Chun Lin, Sidhanth Mohanty, Ryan O'Donnell, and Rachel Yun Zhang. Explicit Two-Sided Vertex Expanders beyond the Spectral Barrier. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, STOC '25, pages 833–842, New York, NY, USA, June 2025. Association for Computing Machinery. doi:10.1145/3717823.3718241.

12 Jun-Ting Hsieh, Alexander Lubotzky, Sidhanth Mohanty, Assaf Reiner, and Rachel Yun Zhang. Explicit Lossless Vertex Expanders. In *2025 IEEE 66th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 894–911, December 2025. ISSN: 2575-8454. doi:10.1109/FOCS63196.2025.00046.

13 Jun-Ting Hsieh, Theo McKenzie, Sidhanth Mohanty, and Pedro Paredes. Explicit Two-Sided Unique-Neighbor Expanders. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, pages 788–799, New York, NY, USA, June 2024. Association for Computing Machinery. doi:10.1145/3618260.3649705.

14 Nabil Kahale. Eigenvalues and expansion of regular graphs. *J. ACM*, 42(5):1091–1106, September 1995. doi:10.1145/210118.210136.

15 Itay Kalev and Amnon Ta-Shma. Unbalanced Expanders from Multiplicity Codes. In *In Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2022)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.APPROX/RANDOM.2022.12.

16 Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. *J. ACM*, 61(5):28:1–28:20, September 2014. doi:10.1145/2629416.

17 Ting-Chun Lin and Min-Hsiu Hsieh. Good quantum LDPC codes with linear time decoder from lossless expanders, March 2022. arXiv:2203.03581 [quant-ph]. doi:10.48550/arXiv.2203.03581.

18 Theo McKenzie and Sidhanth Mohanty. High-girth near-ramanujan graphs with lossy vertex expansion. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 96:1–96:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ICALP.2021.96.

19 Joseph Fels Ritt. *Differential algebra*, volume 33. American Mathematical Soc., 1950.

20 M. Sipser and D.A. Spielman. Expander codes. *IEEE Transactions on Information Theory*, 42(6):1710–1722, November 1996. Conference Name: IEEE Transactions on Information Theory. doi:10.1109/18.556667.

21 Amnon Ta-Shma and Christopher Umans. Better lossless condensers through derandomized curve samplers. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 177–186, October 2006. doi:10.1109/FOCS.2006.18.

22 Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Lossless Condensers, Unbalanced Expanders, And Extractors. *Combinatorica*, 27(2):213–240, March 2007. doi:10.1007/s00493-007-0053-2.

23 Eli Upfal and Avi Wigderson. How to share memory in a distributed system. *J. ACM*, 34(1):116–127, January 1987. doi:10.1145/7531.7926.

A Explicitness

▷ Claim 47. The KT graph G as defined in Definition 20 is explicit, i.e., the left neighborhood function $\Gamma_{\rightarrow} : \mathbb{F}_q^n \times \mathbb{F}_q \to \mathbb{F}_q^{s+2}$ and right neighborhood function $\Gamma_{\leftarrow} : \mathbb{F}_q^{s+2} \times \mathbb{F}_q^{n-(s+1)} \to \mathbb{F}_q^n$ can be computed in $\text{poly}(n, \log(q))$ time.

Proof. To compute $\Gamma_{\rightarrow}(f, y)$, we treat f as an element of $\mathbb{F}_q[X]$ of degree at most $n-1$, and map it to $(y, f^{(0)}(y), \ldots, f^{(s)}(y))$. We can compute derivatives of f and evaluate it at y in time $\text{poly}(n, \log(q))$ and hence explicitly compute $\Gamma_{\rightarrow}(f, y)$.

To compute $\Gamma_{\leftarrow}(z, t)$, we proceed as follows. Let $z = (y, w)$ where $y \in \mathbb{F}_q$ and $w \in \mathbb{F}_q^{s+1}$. Then, we need to find f such that $\Gamma_{\rightarrow}(f, y) = z$. Define $\psi_y : \mathbb{F}_q^n \to \mathbb{F}_q^{s+1}$ as $\psi_y(f) = (f^{(0)}(y), \ldots, f^{(s)}(y))$. By Lemma 25, ψ_y is a full rank $\mathbb{F}_q$-linear map. As $n > s+1$, kernel of ψ_y has dimension $n-(s+1) > 0$. By considering the matrix associated with ψ_y and using standard linear algebra algorithms, we construct an injective linear map $K_y : \mathbb{F}_q^{n-(s+1)} \to \mathbb{F}_q^n$ in $\text{poly}(n, \log(q))$ time such that the image of K_y is exactly the kernel of ψ_y. Furthermore, using Gaussian elimination on the matrix associated with ψ_y, we can, in $\text{poly}(n, \log(q))$ time, find some $g \in \mathbb{F}_q^n$ such that $\psi_y(g) = w$. Finally, we let $\Gamma_{\leftarrow}(z, t) = K_y(t) + g$. By linearity of ψ_y, we have that $\psi_y(K_y(t) + g) = \psi_y(g) = w$. As K_y is injective, for a fixed z, our computed function maps different t to different outputs as desired. ◁

Note that given any $n, k, \varepsilon, \alpha$, Lemma 31 sets $q = \text{poly}(n, k, 1/\varepsilon)$, so we can deterministically find such a prime q satisfying the requirements in $\text{poly}(n, 1/\varepsilon)$ time as well.

B The [10] Graph is Not Right Regular

One may naturally try to show that the predecessor to the KT graph, the [10] graph, is also a two-sided lossless expander. However, it turns out that the [10] graph is not even right regular. To see why, we give the definition of the [10] graph which is similar to the KT graph.

▶ **Definition 48** (The GUV graph, [10]). *Let $q, n, m, h \in \mathbb{N}$ be such that q is a prime power greater than h, characteristic of the finite field $\mathbb{F}_q \geq n$ and $m < n$. We define $G = (L \sqcup R, E)$ where $L = \mathbb{F}_q^n \cong \mathbb{F}_q[x]/(z(x))$ for some irreducible polynomial $z(x) \in \mathbb{F}_q(x)$ of degree n and $R = \mathbb{F}_q^{m+1}$. The left degree is q and for any $f \in \mathbb{F}_q[x]/(z(x))$ and $y \in \mathbb{F}_q$, the y'th neighbor of f is defined as*

$$\Gamma_{\rightarrow}(f, y) = \left(y, f(y), (f^h \bmod z(x))(y), (f^{h^2} \bmod z(x))(y), \ldots, (f^{h^{m-1}} \bmod z(x))(y)\right).$$

Our proof of right-regularity - Lemma 25 - relied on the fact that the map $\psi_y(f) \mapsto (f^{(0)}(y), \ldots, f^{(s)}(y))$ is full rank over $\mathbb{F}_q$. The analogous GUV map defined as $\varphi_y(f) = (f(y), f^h(y), \ldots, f^{h^{m-1}}(y))$ does not have this property because of two issues. First, it is not necessarily linear over $\mathbb{F}_q$, although it can be made linear over $\mathbb{F}_2$ when q is a power of 2. Second, even over $\mathbb{F}_2$, it does not necessarily have full rank, meaning we cannot guarantee right regularity.

Simulations bear this out. The GUV graph with $q = 2^4$, $n = 4$, $m = 2$, and $h = 2$ has 3072 right vertices with degree 256, 64 with degree 4096, and 960 isolated vertices. For more examples of parameter settings where the GUV graph is not right-regular, we invite the reader to run simulations with our code at `https://github.com/mjguru/GUV-Expander`.

Towards Tight Bounds for Testing k-Colorability

Nick Kushnir
School of Mathematics, Tel Aviv University, Israel

Asaf Shapira
School of Mathematics, Tel Aviv University, Israel

Abstract

Determining the sample complexity for testing k-colorability is perhaps the most well studied problem in property testing. It was (implicitly) introduced almost 50 years ago by Bollobás, Erdős, Simonovits and Szemerédi, who used the regularity lemma in order to give a tower-type bound for this problem. This bound has been successively improved in a long list of works, bringing the state-of-the-art bounds to lie between $\Omega(1/\varepsilon)$ and $O((k/\varepsilon)\log^2(1/\varepsilon))$. We obtain the following new results:

(i) Our first main result is an improved $O((k/\varepsilon)\log(1/\varepsilon))$ upper bound, bringing the sample complexity closer to "truly" linear in ε. To prove this result, we improve upon a variant of the container method, introduced recently in the breakthrough paper of Blais and Seth.

(ii) Perhaps the most important gap in our understanding of this problem is that while the best known upper bound increases with k, the lower bound is independent of k. Our second main result fills this gap by providing a new $\Omega(\frac{\log k}{\varepsilon})$ lower bound, improving upon a result of Alon and Krivelevich from 2002. We conjecture that this is the true sample complexity of k-colorability.

2012 ACM Subject Classification Mathematics of computing → Graph theory; Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases k-colorability, property testing, sample complexity, graph property testing, random graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.35

Category RANDOM

Funding *Asaf Shapira*: Supported in part by ERC Consolidator Grant 863438.

1 Introduction

Given a graph $G = (V, E)$, we use $\chi(G)$ to denote its *chromatic number*, that is, the smallest integer k so that V can be colored with the colors $\{1, \ldots, k\}$ so that for each edge $(u, v) \in E$ the vertices u, v are colored differently. Computing $\chi(G)$, or even just determining if $\chi(G) \leq 3$, is of course one of the classical NP-hard problems. One of the best "combinatorial evidence" for the hardness of determining $\chi(G)$ is a classical result of Erdős [12], which states that for arbitrary k and n, there are n-vertex graphs of chromatic number k in which *every* set of $\Omega(n)$ vertices induces a 3-colorable graph. Qualitatively, this means that the chromatic number of a graph lacks any relation between its global and local aspects. Suppose now that we strengthen our global assumption on G, and instead of just assuming that it has large chromatic number, we further assume that it is "far" from having small chromatic number. Can we then show that G has a subgraph on a small number of vertices which witnesses this fact? Interestingly, the answer in this case is dramatically different. We now make this statement more precise.


© Nick Kushnir and Asaf Shapira;
licensed under Creative Commons License CC-BY 4.0

For a fixed $\varepsilon > 0$ we say that G is ε-far from being k-colorable, if one should remove from G at least εn^2 edges in order to obtain a graph which is k-colorable[1]. A result of Bollobás, Erdős, Simonovits and Szemerédi [10] from 1978 states that if G is ε-far from being 2-colorable then G contains a set of vertices of size $C(\varepsilon)$ which spans a non 2-colorable graph. This result was later extended by Rödl and Duke to arbitrary fixed k. Both proofs [10, 25] relied on Szemerédi's regularity lemma [29], therefore they only supplied tower-type bounds for $C(\varepsilon)$. Interestingly, both proofs in fact show that a random sample of $C(\varepsilon)$ vertices spans a non-k-colorable subgraph with probability at least 2/3. Let us then define $S_k(\varepsilon)$ to be the smallest q such that if G is ε-far from being k-colorable then a sample of q vertices spans a non-k-colorable subgraph with probability at least 2/3. We call $S_k(\varepsilon)$ the *sample complexity* of k-colorability.

We pause our discussion for a moment and observe that the fact that $S_k(\varepsilon)$ is independent of n has an immediate surprising algorithmic implication. A randomized algorithm is said to be an ε-tester for k-colorability if it accepts k-colorable graphs with probability at least 2/3 and rejects those that are ε-far from being k-colorable with probability at least 2/3. It is now easy to see that we can obtain an ε-tester for k-colorability which simply samples $S_k(\varepsilon)$ vertices, and checks[2] if the subgraph they induce is k-colorable. This observation was made explicit by Goldreich, Goldwasser and Ron [17] in their seminal paper which introduced[3] the area of property testing, and where similar algorithms were designed for more general partition properties. See [5, 16] for more details on property testing. The take-home message here is that besides the purely graph theoretic interest in understanding $S_k(\varepsilon)$, an additional motivation stems from its algorithmic application.

The study of the sample complexity of k-colorability took two dramatic turns in the mid 90's. First, Frieze and Kannan [14, 15] made the crucial observation that in order to study $S_k(\varepsilon)$ one does not need the full power of Szemerédi's regularity lemma [29], and that a weaker regularity lemma suffices. This observation brought down the sample complexity to $2^{(k/\varepsilon)^{O(1)}}$, already making a significant improvement over the regularity-based tower-type bound of Rödl and Duke [25]. Independently of [14, 15], Goldreich, Goldwasser and Ron [17] obtained the first polynomial bound for $S_k(\varepsilon)$. They did so by completely doing away with the regularity lemma method, showing that $S_k(\varepsilon) = \tilde{O}(k^2/\varepsilon^3)$. This bound was later improved by Alon and Krivelevich [1] to $\tilde{O}(k/\varepsilon^2)$. A significant breakthrough was obtained by Sohler [28] who improved the dependence on ε to near linear, giving an upper bound of $\tilde{O}(k^6/\varepsilon)$. In a recent breakthrough, Blais and Seth [7] further improved this, showing that $S_k(\varepsilon) = O((k/\varepsilon)\log^2(1/\varepsilon))$. In terms of lower bounds, Alon and Krivelevich [1] proved that $S_k(\varepsilon) = \Omega(1/\varepsilon)$.

As is evident, determining the correct order of $S_k(\varepsilon)$ has long served as a testbed for developing many of the most important techniques that became ubiquitous in the area of graph property testing. Consequently, resolving this problem was raised as an open problem

---

[1] Observe that every graph can be made k-colorable by partitioning the vertices into k sets of equal size and removing the edges within these sets. We thus remove at most n^2/k, hence we can always assume that $k \leq 1/\varepsilon$.

[2] Note that this algorithm actually accepts with probability 1 every k-colorable input.

[3] We should point out that [17] used the notion of *query complexity* which is the number of edges we need to sample from G in order to design an ε-tester for k-colorability. Clearly the query complexity is at most quadratic in $S_k(\varepsilon)$, since we can simply query all edges inside the sampled set of vertices. Interestingly, there is a theorem of Goldreich and Trevisan [19] stating that this quadratic gap actually holds for every graph property, and a recent theorem of Goldreich and Wigderson [20] stating that it is sometimes unavoidable. See [21, 18] for further discussion. Since almost all works on k-colorability used the notion of sample complexity, we also use it here.

in several papers[4], for example in 2002 by Alon and Krivelevich [1] and more recently by Blais and Seth [7]. Observe that a striking inconsistency has persisted for the past 20 years: while all upper bounds for $S_k(\varepsilon)$ increase with k, the only known lower bound is independent of k. Our first main result in this paper fills this qualitative gap, improving upon the lower bound obtained in 2002 by Alon and Krivelevich [1].

▶ **Theorem 1.1.** *The sample complexity of k-colorability satisfies $S_k(\varepsilon) = \Omega(\frac{\log k}{\varepsilon})$.*

The main idea of the proof of Theorem 1.1 is the following. As we mentioned above Erdős [12] showed that there is a graph of chromatic number k in which every set of vertices of linear size is 3-colorable. Alon and Krivelevich [1] showed that by taking a blowup of such a graph one can obtain an $\Omega(1/\varepsilon)$ lower bound. The parameters in Erdős's example are tight if one insists that *every* set of vertices of linear size has small chromatic number. Our main innovation here is that in the setting of testing it is enough that most sets satisfy this condition, which allows us to use a theorem of Łuczak [23] on the chromatic number of sparse random graphs.

Our second main result in this paper improves the state of the art upper bound as follows.

▶ **Theorem 1.2.** *The sample complexity of k-colorability satisfies $S_k(\varepsilon) = O(\frac{k}{\varepsilon}\log\frac{1}{\varepsilon})$.*

The proof of Theorem 1.2 builds on the approach introduced in the recent breakthrough of Blais and Seth [7, 6], which itself is a variant of the *Container Method* in extremal combinatorics. This method very roughly states that if the edges of a (hyper)graph are reasonably well distributed then there is a small collection of vertex subsets $C_1, \ldots, C_t$ called *containers*, so that every independent set is a subset of one of these containers. The main advantage is that this lemma allows one to improve upon naive union bounds since instead of having to union bound over all independent sets, one only needs to consider the sets C_i. This method has its origins in the works of Kleitman and Winston [22] and Sapozhenko [26] who considered containers in graphs. A major breakthrough was obtained about 10 years ago by Balogh, Morris, and Samotij [4] and by Saxton and Thomason [27] who extended this method also to hypergraphs, allowing many more applications of this method. Blais and Seth [7, 6] found a clever way of applying this method in the setting of testing graph properties. As in [4, 27] the main challenge is to design an algorithm[5] which receives the graph and constructs the containers. Our main contribution here is in improving the analysis of the algorithm presented in [6]. More concretely, in Lemma 3.10 we remove a $\log 1/\varepsilon$ factor from the bound obtained in [6] for the same lemma (Lemma 6[6] in [6]), which results in the improved upper bound in Theorem 1.2.

The two main differences between our analysis here and the ones in [7, 6] are the following: First, we use the reduction of [6] which reduces testing k-colorability to finding a single independent set in an appropriate graph (the one in [7] requires k independent sets which is more costly), but we run on this graph the container algorithm of [7], which is much simpler to analyse and adapt to our needs. The second idea is to break the analysis of the container

[4] It is worth mentioning that some papers [8, 9] focused on the special case of determining $S_2(\varepsilon)$ (which might be simpler given the simple structure of bipartite graphs) while other papers [2, 6, 11] studied the more general problem of hypergraph colorability.

[5] We stress that by "algorithm" we refer to the procedure that (explicitly) constructs the containers, and not to any algorithmic aspect of testing k-colorability.

[6] The result of [6] actually applies to more general hypergraph coloring properties. Our improved bound can be easily extended to that setting, but we chose to focus on the graph setting in order to simplify the proofs.

algorithm of [7] into Claims 3.3, 3.4 and 3.5, which allows us to make the crucial observation in Claim 3.4. This claim allows us to save a $\log \frac{1}{\varepsilon}$ factor in the edge counts at the end of the proof of Lemma 3.10. To summarize, we combine the ideas presented in [7, 6] with the crucial observation in Claim 3.4.

Our new lower/upper bounds suggest the natural question: what is the true asymptotic behaviour of $S_k(\varepsilon)$? While the k term and the $\log 1/\varepsilon$ term in the upper bound seem like an artifact of the proof, it appears as if the $\log k$ term in the lower bound is much harder to improve upon. We thus suspect that the following holds.

▶ **Conjecture 1.3.** *The sample complexity of k-colorability satisfies $S_k(\varepsilon) = \Theta(\frac{\log k}{\varepsilon})$.*

For what it is worth, our feeling is that it will be hard to remove the final $\log \frac{1}{\varepsilon}$ factor from the upper bound on $S_k(\varepsilon)$ using the method of [7, 6], but that it should be possible to improve the dependence on k (perhaps even all the way to the optimal/conjectured $\log k$) using this method.

Two intermediate goals towards Conjecture 1.3 are to prove an upper bound of $C(k)/\varepsilon$ (for some arbitrary constant $C(k)$) and an upper bound of $\frac{(\log k)^C}{\varepsilon} \log \frac{1}{\varepsilon}$. Observe that a bound of the form $O(\log k/\varepsilon^C)$ (with $C = 7$) is known [3, 13], but is inferior to the known $O(\frac{k}{\varepsilon} \log^{O(1)} \frac{1}{\varepsilon})$ bounds as long as $C \geq 2$ since, as we mentioned earlier, we can assume that $k \leq 1/\varepsilon$. It would also be very interesting to investigate the sample complexity of *tolerantly* testing k-colorability [24], namely, estimating how far is an input from being k-colorable. While this can be done with sample complexity polynomial in $1/\varepsilon$ using the results of [17], determining the sample complexity of this problem seems very challenging.

Paper overview

In Section 2 we prove Theorem 1.1. We first state and prove Lemma 2.3 which is the main lemma in the section, and then use it to prove the lower bound. In Section 3 we prove Theorem 1.2 using the container method.

2 Proof of Theorem 1.1

Our proof of Theorem 1.1 will rely on the following theorem of Łuczak [23] regarding the chromatic number of $G(n, p)$.

▶ **Theorem 2.1.** *There exists a constant d_0 such that if $d = d(n) = n \cdot p(n) > d_0$ and $p(n) = o(1)$, then with probability $1 - o(1)$*

$$\chi(G(n,p)) \leq \frac{np}{\log np}$$

Łuczak [23] actually obtains the much stronger bound $\chi(G(n,p)) < (\frac{1}{2} + o(1))np/\log np$ but the above result suffices for our purposes here. We will also need the following basic Chernoff bound.

▶ **Lemma 2.2.** *Given $p_1, \ldots, p_n \in [0,1]$ and mutually independent random variables $X_1, \ldots, X_n$, taking values in $\{0,1\}$, with $\mathbb{P}[X_i = 1] = p_i$, denote $p = (p_1 + \cdots + p_n)/n$ and $X = X_1 + \cdots + X_n$. Then for every $0 < a \leq pn$*

$$\mathbb{P}[X - \mathbb{E}[X] < -a] < e^{-a^2/2pn}$$

In order to prove Theorem 1.1, we will first prove the following lemma.

▶ **Lemma 2.3.** *There is k_0 so that for every $k > k_0$, and every large enough m there is an m-vertex graph H satisfying:*

- **(i)** *For every $1 \leq r \leq m/80$, a uniformly sampled subset of r vertices of H spans a k-colorable graph with probability at least $\frac{9}{10}$.*
- **(ii)** *One needs to remove from H at least $m \log k$ edges in order to make it k-colorable.*

Proof. Set $p = 40k \log k/m$. To prove the lemma, we will show that for all large enough m, the random graph $H = G(m,p)$ satisfies each of the two desired properties with probability at least $3/4$, and hence there is a graph satisfying both properties.

We first prove that property (i) holds with probability at least $3/4$ with respect to $r = m/80$. Fix a set R of $m/80$ vertices of $H = G(m,p)$. Due to the independence of the $G(m,p)$ model, we can think of the induced subgraph $H[R]$ as generated by $G(m/80,p)$. Thus, by Theorem 2.1, with the parameters $n = m/80, p = 40k \log k/m$, we deduce that for all large enough k[7] and all large enough m, with probability at least 0.99 we have

$$\chi(R) \leq \frac{np}{\log np} = \frac{k \log k}{2 \log\left(\frac{k \log k}{2}\right)} < k \; .$$

Let $X = X(H)$ denote the number of subgraphs of $H = G(m,p)$ of size $m/80$ that are not k-colorable. By the above assertion we have $\mathbb{E}[X] \leq 0.01 \cdot \binom{m}{m/80}$. Hence, by Markov's inequality

$$\mathbb{P}\left[X > 0.1\binom{m}{m/80}\right] < 1/4 \; .$$

Therefore, with probability at least $3/4$, at most $0.1\binom{m}{m/80}$ of the subsets of H of size $m/80$ span a non-k-colorable subgraph, which is exactly what property (i) states for $r = m/80$. Consider now any $r' < r$. We can sample a random set of size r' by first picking random set of size r and then picking from it a random set of size r'. By the above argument, the random set of size r spans a k-colorable graph with probability at least $9/10$, and since a subgraph of a k-colorable graph is also k-colorable, a random set of size r' is also k-colorable with probability at least $9/10$.

We now prove that property (ii) holds with probability at least $3/4$. Towards this, we first prove that with probability at least $3/4$ every set of vertices R in H of size $|R| = r \geq \frac{m}{2k}$ contains at least $2r \log k$ edges. Let us then fix a subset $R \subset V(H)$ of size r. From Lemma 2.2, with $n = \binom{r}{2}$, all $p_i = p$ and $a = \binom{r}{2}p - 2r \log k$, we have

$$\begin{aligned}
\mathbb{P}[|E(G[R])| < 2r \log k] &< e^{-\left(\binom{r}{2}p - 2r \log k\right)^2 / 2\binom{r}{2}p} \\
&< e^{-(rp - 5 \log k)^2/(4p)} \\
&< e^{-pr^2/10} \; .
\end{aligned}$$

[7] We need k to be large enough so that $pm = \frac{1}{2}k \log k$ is larger than d_0 in Theorem 2.1.

Hence, the probability that there is a subset R of size $m/2k \leq r \leq m$ violating the above property is at most

$$\begin{aligned}\sum_{m/2k\leq r\leq m}\binom{m}{r}e^{-pr^2/10} &\leq \sum_{m/2k\leq r\leq m}(2ek)^r\cdot e^{-pr^2/10}\\ &\leq \sum_{m/2k\leq r\leq m}e^{r(\ln(2ek)-pr/10)}\\ &\leq \sum_{m/2k\leq r\leq m}e^{r(\ln(2ek)-2\log k)}\\ &\leq \sum_{m/2k\leq r\leq m}e^{r(-\log k)}=o(1)\end{aligned}$$

We now prove that if the condition in the previous paragraph holds, then item (ii) holds as well, and hence it holds with probability at least 3/4. Indeed, take any partition of $V(H)$ into k sets $C_1,\ldots,C_k$. Since at most half of H's vertices belong to sets C_i of size at most $m/2k$ and since any C_i of size at least $m/2k$ contains at least $2|C_i|\log k$ edges, we conclude that H contains at least $m\log k$ edges within the sets $C_1,\ldots,C_k$. Since this is true for any partition into k sets, at least $m\log k$ edges need to be removed from H to make it k-colorable. ◀

Proof (of Theorem 1.1). Let k_0 be the constant in Lemma 2.3. We may clearly assume that $k > k_0$ since we can either absorb this into the constant in the $\Omega(\frac{\log k}{\varepsilon})$ bound or alternatively just use the $\Omega(\frac{1}{\varepsilon})$ lower bound of [1] which holds for all k. Given $k > k_0$ and all small enough $\varepsilon > 0$, we will show that there are infinitely many $n \in \mathbb{N}$ for which there is a graph G which is ε-far from being k-colorable, and yet a randomly chosen set of $\Omega(\frac{\log k}{\varepsilon})$ vertices spans a k-colorable graph with probability at least 9/10. Set

$$m = \frac{\ln k}{\varepsilon}\ . \tag{1}$$

Since we may assume that ε is small enough, we can assume that m is large enough for Lemma 2.3 to hold. Let H be the graph guaranteed by the lemma. For each integer n, let G be the n/m-blowup of H, that is, the graph obtained by replacing each vertex v of H with an independent set I_v of size n/m, and every edge (u,v) of H with a complete bipartite graph connecting the sets I_u, I_v. We will now show that G satisfies the two properties above, namely, that it is ε-far from being k-colorable, and that a sample of $s = m/80 = \Omega(\frac{\log k}{\varepsilon})$ of vertices spans a k-colorable graph with probability at least 9/10.

Let us number the vertices of H as $1,\ldots,m$ and the vertex sets that replaced them in G as $I_1,\ldots,I_m$. Given a set of vertices S in G, let $S_j = S \cap I_j$ and let S_H be the set of vertices $1 \leq j \leq m$ of H for which $S_j \neq \emptyset$. It is now easy to see that for each S, if the graph $H[S_H]$ is k-colorable then so is $G[S]$. Indeed, we can color all the vertices of I_j in G using the color assigned to vertex j in H. We also observe that by symmetry, if we randomly select a set S in G of size s, then for every $s' < s$, all sets of size s' are equally likely to form the set S_H. We deduce that conditioned on the size of S_H, the set S_H is a uniform subset of H of size at most $m/80$. By Lemma 2.3 a uniform set S_H of any fixed size at most $m/80$ in H is k-colorable with probability at least 9/10. Hence, if we randomly pick a set S of size $m/80$ with probability at least 9/10 the graph $H[S_H]$ is k-colorable. But from our observation above, this means that with probability at least 9/10 the graph $G[S]$ is also k-colorable.

We now prove that G is ε-far from being k-colorable. Take any partition of $V(G)$ into k color sets $C_1, \ldots, C_k$, and let M denote the number of monochromatic edges of this partition, namely, the number of edges within these k sets. Let use this k-coloring in order to (randomly) k-color H as follows: assign color q to a vertex u with probability $|C_q \cap I_u|/|I_u|$. The expected number of monochromatic edges of H in such a coloring is precisely

$$\begin{aligned}\sum_{(u,v)\in E(H)} \sum_{q=1}^{k} \frac{|C_q \cap I_u|}{|I_u|} \frac{|C_q \cap I_v|}{|I_v|} &= \frac{m^2}{n^2} \sum_{q=1}^{k} \sum_{(u,v)\in E(H)} |C_q \cap I_u||C_q \cap I_v| \\ &= \frac{m^2}{n^2} \sum_{q=1}^{k} |(x,y) \in E(G) : x, y \in C_q| \\ &= \frac{M \cdot m^2}{n^2} \,.\end{aligned}$$

By Item (ii) of Lemma 2.3, every k-coloring of H creates at least $m \log k$ monochromatic edges, implying that $\frac{M \cdot m^2}{n^2} \geq m \log k$ and recalling (1) this means that $M \geq \varepsilon n^2$. We conclude that every k-coloring of G creates at least εn^2 monochromatic edges, implying that G is ε-far from k-colorable. ◀

3 Proof of Theorem 1.2

As we mentioned in Section 1, our proof relies on the container method. Algorithm-1 described below receives as input an independent set I in a graph G and produces a sequence $C_1, C_2, \ldots$ of supersets of I, which we call "containers", together with a sequence $F_1, F_2, \ldots$ of subsets of I, which we call "fingerprints". As the name suggests, each fingerprint F_i determines the container C_i. The algorithm constructs these two sequences using a (surprisingly) simple greedy algorithm. Starting with $F_0 = \emptyset$ and $C_0 = V(G)$, in each iteration t the algorithm picks a new vertex $v_t \in I \setminus F_{t-1}$ to be added to the fingerprint. It does so by taking v_t to be the vertex in $I \setminus F_{t-1}$ of the highest degree in $G[C_{t-1}]$ (the graph induced by G on C_{t-1}). It then adds v_t to F_{t-1} to obtain F_t. It also removes all neighbors of v_t from C_{t-1} (none of them belongs to I since it is an independent set) as well as all vertices of C_{t-1} of degree larger than that of v_t (none of them belongs to I by choice of v_t). Note that the algorithm terminates after $|I|$ steps but we further set $C_t = F_t = I$ for $t > |I|$.

Algorithm 1 Fingerprint And Container Generator.

Input: A graph $H = (V, E)$ and an independent set $I \subseteq V$

Initialize $F_0 \leftarrow \emptyset$ and $C_0 \leftarrow V(H)$;
for $t = 1, 2, \ldots, |I|$ **do**
  $v_t \leftarrow$ the vertex in $I \setminus F_{t-1}$ that has the largest degree in $H[C_{t-1}]$;
  `// remove all vertices with higher degree than` v_t
  $X \leftarrow \{w \in C_{t-1} : \deg_{H[C_{t-1}]}(w) > \deg_{H[C_{t-1}]}(v_t)\}$;
  `// remove all neighbors of` v_t
  $Y \leftarrow \{w \in V : \{w, v_t\} \in E\}$;
  `// Add` v_t `to the fingerprint`
  $F_t \leftarrow F_{t-1} \cup \{v_t\}$;
  `// Update container set by removing` X `and` Y
  $C_t \leftarrow C_{t-1} \setminus (X \cup Y)$;
Return $F_1, \ldots, F_{|I|}$ and $C_1, \ldots, C_{|I|}$;

3.1 Basic Properties

We now prove several useful properties regarding Algorithm 1.

▶ **Definition 3.1.** *For a graph $H = (V, E)$ and an independent set ($I \subseteq V$), we write $C_t(I)$ for the container obtained after t iterations of the Algorithm when the algorithm is run with input I on the graph H. Similarly, $F_t(I)$ denotes the corresponding fingerprint after t iterations.*

By construction, we have

$$F_1 \subseteq F_2 \subseteq \cdots \subseteq F_{|I|} = I \subseteq C_{|I|} \subseteq C_{|I|-1} \subseteq \cdots \subseteq C_1 \subseteq C_0 \tag{2}$$

Let us state more useful properties of these sequences. Note that the construction of the containers and the fingerprints given the independent set I guarantees that the $C_t(I)$ equals $C_t(F_t(I))$. This is stated in the following claim:

▷ Claim 3.2. For any graph $H = (V, E)$ and any independent set I in H, and any t, the fingerprint $F_t(I)$ and container $C_t(I)$ of I satisfy

$$C_t\big(F_t(I)\big) = C_t(I).$$

Proof. Fix t to be the iteration index. We will prove that $C_t\big(F_t(I)\big) = C_t(I)$. We will prove by induction on the iteration index i that for every $i \leq t$ we have $C_i\big(F_t(I)\big) = C_i(I)$. For $i = 0$ the statement is trivial. Let us assume that the statement is true for i and prove it for $i + 1$. From the inductive assumption we know that $C_i\big(F_t(I)\big) = C_i(I)$. Consider the vertex v_{i+1} that was chosen in the $(i + 1) - th$ iteration of the algorithm (running on input I). By construction, the degree of v_{i+1} in $C_i(I)$ is the highest from $I \backslash F_i$. In particular, the degree is also the highest from the following set $F_t \backslash F_i$ in the graph $C_i(I)$ and hence also in $C_i\big(F_t(I)\big)$, and so it is chosen as the $i + 1$ vertex when the algorithm is running on $F_t(I)$ as well. As the current C_i are the same in both algorithms, and so is the chosen vertex in the $(i + 1)$-th iteration, it follows that $C_{i+1}\big(F_t(I)\big) = C_{i+1}(I)$. ◁

▷ Claim 3.3. Let $H = (V, E)$ be a graph, let I be an independent set in H, and let $0 \leq t \leq |I|$. Denote by Δ_t the maximal degree in C_t. Then, Δ_t is weakly decreasing w.r.t t.

Proof. Follows immediately from the fact that $C_t \subseteq C_{t-1}$ as stated in (2). ◁

▷ Claim 3.4. Let $H = (V, E)$ be a graph, let I be an independent set in H, let $1 \leq t \leq |I|$ and let Δ_t denote the maximum degree of $H[C_t(I)]$. Then $\Delta_t \leq |C_{t-1} \backslash C_t|$.

Proof. Suppose v_t is the vertex in $I \backslash F_{t-1}$ that has the largest degree in $H[C_{t-1}]$. Let $d(v_t)$ denote the degree of vertex v_t in $H[C_{t-1}]$. Then $|C_{t-1} \backslash C_t| \geq d(v_t)$ since Algorithm-1 removes from C_{t-1} the $d(v_t)$ neighbors of v_t in C_{t-1}. On the other hand $\Delta_t \leq d(v_t)$ since Algorithm-1 removes from C_{t-1} all vertices of degree larger than $d(v_t)$. ◁

▷ Claim 3.5. Let $H = (V, E)$ be a graph, let I be an independent set in H, and let $1 \leq t \leq |I|$. Then the maximum degree of $H[C_t]$ is at most $|V|/t$.

Proof. Fix $1 \leq t \leq |I|$. As stated in (2), we have $C_t \subseteq C_{t-1} \subseteq \cdots \subseteq C_1 \subseteq C_0$. Hence,

$$|V| \geq |C_0| - |C_t| = \sum_{i=1}^{t} \big(|C_{i-1}| - |C_i|\big) = \sum_{i=1}^{t} |C_{i-1} \setminus C_i| \,.$$

Therefore, there is some index $j \in [t]$ such that $|C_{j-1} \backslash C_j| \leq |V|/t$. Thus, in the notation of Claim 3.4, we have $\Delta_j \leq |V|/t$. By Claims 3.3 and 3.4 we have also $\Delta_t \leq |V|/t$. ◁

3.2 Graph Construction

A k-coloring of a graph G is given by k independent sets corresponding to the k color classes. Therefore, if we want to apply the container algorithm to such a description, we have to apply it k times, once for each of the k independent sets. We now describe a simple way to encode the k-colorings of a graph G using another graph H. The advantage is that now every k-coloring of G corresponds to a single independent set in H. This will simplify the analysis in the next two subsections.

▶ **Definition 3.6.** *Given a graph $G = (V, E)$ and an integer k, we define the graph $H_{G,k} = (\bar{V}, \bar{E})$ as follows. If the vertices of G are $V = \{v_1, \ldots, v_n\}$ then $H_{G,k}$ has the following $k \cdot n$ vertices: for each vertex $v_i \in V$, the vertex set $\bar{V}$ contains k vertices $v_{i,1}, \ldots, v_{i,k}$. For what follows, let $S_i = \{v_{i,1}, \ldots, v_{i,k}\}$. The edges of $H_{G,k}$ are the following: for every $v_i \in V$ we put a complete graph on S_v, and for every edge (v_i, v_j) of G and every $1 \leq q \leq k$ we have and edge $(v_{i,q}, v_{j,q})$.*

One should think of vertex $v_{i,q}$ as indicating the fact that v_i is colored q. The following follows immediately from the definition above.

▶ **Fact 3.7.** *Given a graph $G = (V, E)$ and an integer k let $H_{G,k} = (\bar{V}, \bar{E})$ be the graph from Definition 3.6. Suppose $S = \{v_1, \ldots, v_s\} \subseteq V(G)$ is such that $G[S]$ is k-colorable with k-coloring $\phi : S \mapsto [k]$. Then the set $S_\phi = \{v_{1,\phi(v_1)}, \ldots, v_{s,\phi(v_s)}\}$ is an independent set in $H_{G,k}$.*

▶ **Definition 3.8.** *Given a graph $G = (V, E)$ and an integer k, let $H_{G,k} = (\bar{V}, \bar{E})$ be the graph from Definition 3.6. If $U \subset \bar{V}$ then we use vert(U) to denote the set of vertices v for which $S_v \cap U \neq \emptyset$, that is, the set of vertices $v_i \in V(G)$ so that at least one vertex $v_{i,q}$ is in U.*

▶ **Observation 3.9.** *Given a graph $G = (V, E)$ and an integer k, let $H_{G,k} = (\bar{V}, \bar{E})$ be the graph from Definition 3.6. Suppose $U \subset \bar{V}$ satisfies vert$(U) = V(G)$ and $|U| = n$ (i.e. so that $|U \cap S_v| = 1$ for every $v \in V(G)$). Then U encodes a coloring $\phi : V \to [k]$ on the vertices of G. Furthermore, the number of edges in $H[U]$ is exactly the number of monochromatic edges in G w.r.t the coloring ϕ.*

Proof. Let $\phi : V \mapsto [k]$ be such that $\phi(v_i) = q$ if $v_{i,q} \in U$. It is then clear from definition of $H_{G,k}$ that every monochromatic edge created by ϕ corresponds to precisely one edge of $H[U]$. ◀

3.3 A New Container Lemma

The next lemma is the key tool we need for the proof of Theorem 1.2. It improves upon a similar lemma from [6] by a $\log 1/\varepsilon$ factor[8].

▶ **Lemma 3.10.** *Given a graph $G = (V, E)$ and an integer k let $H_{G,k} = (\bar{V}, \bar{E})$ be the graph from Definition 3.6. Suppose that for some $\varepsilon > 0$ the graph G is ε-far from being k-colorable. Then, if I is an independent set in $H_{G,k}$ then there exists $1 \leq t \leq 4k/\varepsilon$ such that*

$$|\text{vert}(C_t(I))| \leq \left(1 - \frac{\varepsilon t}{4k}\right) n \,.$$

Here, $C_t(I)$ refers to the container of Algorithm-1.

[8] We remark again that the analogous Lemma 6 in [6] applies also to hypergraph colorability properties, but it is easy to extend our improved bound to this more general setting.

Proof. Assuming by contradiction that for all $1 \leq t \leq 4k/\varepsilon$ we have

$$|\text{vert}(C_t(I))| > \left(1 - \frac{\varepsilon t}{4k}\right) n \ , \tag{3}$$

we will construct a coloring $\phi : V \to [k]$ of the vertices of G, with less than εn^2 monochromatic edges, contradicting the assumption of the lemma. Observe that by Observation 3.9 we can construct such a coloring by finding a set U of size n in $H_{G,k}$ which satisfies $\text{vert}(U) = V$ and which contains less than εn^2 edges. We will construct U as a union of sets $U_0, \ldots, U_{4k/\varepsilon}$ defined as follows.

Let $C_0(I), \ldots, C_{4k/\varepsilon}(I)$ be the containers produced by Algorithm-1 when applied to I in the graph $H_{G,k}$. Recall that $C_t(I) \subseteq C_{t-1}(I)$ so we will consider these containers in reverse order starting from $C_{4k/\varepsilon}(I)$. Given $C_{4k/\varepsilon}(I)$, let $U_{4k/\varepsilon} \subseteq C_{4k/\varepsilon}(I)$ be the following set: for each vertex $v_i \in G$ for which $S_i \cap C_{4k/\varepsilon}(I) \neq \emptyset$, put in $U_{4k/\varepsilon}$ an arbitrarily chosen vertex from $S_i \cap C_{4k/\varepsilon}(I)$.

We now repeat this for every $t = 4k/\varepsilon - 1, \ldots, 0$, constructing a set $U_t \subseteq C_t(I) \setminus C_{t+1}(I)$ as follows: Let R_t be the vertices $v_i \in V(G)$ so that none of the vertices of S_i belongs to $U_{4k/\varepsilon} \cup \ldots \cup U_{t+1}$. Then for each vertex $v_i \in R_t$ for which $S_i \cap (C_t(I) \setminus C_{t+1}(I)) \neq \emptyset$, put in U_t an arbitrarily chosen vertex from $S_i \cap (C_t(I) \setminus C_{t+1}(I))$.

Let $U = U_{4k/\varepsilon} \cup U_{4k/\varepsilon - 1} \cup \ldots \cup U_1 \cup U_0$. Note that our choice or R_t guarantees that U contains at most one vertex from each set S_i. Furthermore, since $C_0 = \bar{V}$ we conclude that U will contain a vertex from each S_i. More precisely, if $S_i \cap C_{4k/\varepsilon} \neq \emptyset$ then we add a vertex of S_i to $U_{4k/\varepsilon}$. Otherwise, if t is the largest[9] integer $1 \leq t \leq 4k/\varepsilon$ so that C_t contains vertices of S_i, then in step t we will add one of the vertices of S_i to U_t. We thus conclude from Observation 3.9 that U encodes a k-coloring of G. Hence, it remains to prove the U contains less than εn^2 edges.

We claim that $H[U_0 \cup U_1 \cup \cdots \cup U_{4k/\varepsilon}]$ contains less than εn^2 edges. Since $U_{4k/\varepsilon} \subseteq C_{4k/\varepsilon}(I)$, then by Claim 3.5 (with $|\bar{V}| = kn$), the maximum degree in the subgraph $H[U_{4k/\varepsilon}]$ is at most $\frac{kn}{4k/\varepsilon} = \frac{\varepsilon}{4} n$. Hence, $H[U_{4k/\varepsilon}]$ has at most $\frac{\varepsilon}{4} n^2$ edges. Now, consider adding to $U_{4k/\varepsilon}$ the vertices of $U_0 \cup \cdots \cup U_{4k/\varepsilon - 1}$ in reverse order. For any $0 \leq t \leq 4k/\varepsilon - 1$, we choose $U_t \subseteq C_t$, hence when adding U_t, the number of edges in $H[U_{4k/\varepsilon} \cup \ldots \cup U_t]$ increases by at most $|U_t| \cdot \Delta_t$ where Δ_t is the maximum degree of $H[C_t(I)]$. We conclude that the number of edges in U is at most

$$\frac{\varepsilon}{4} n^2 + \sum_{t=0}^{4k/\varepsilon - 1} |U_t| \Delta_t \ . \tag{4}$$

Note that our construction guarantees that for any $0 \leq t \leq 4k/\varepsilon - 1$ we have

$$\text{vert}(U_0 \cup \ldots \cup U_t) = V \setminus \text{vert}(C_{t+1}(I)) \ .$$

From our assumption in (3) we know that $|\text{vert}(C_{t+1}(I))| > \left(1 - \frac{\varepsilon(t+1)}{4k}\right) n$, which means that

$$|\text{vert}(U_0 \cup \ldots \cup U_t)| \leq \frac{\varepsilon(t+1)}{4k} n \ .$$

Note also that since (by construction) the sets $\text{vert}(U_0), \ldots, \text{vert}(U_{4k/\varepsilon})$ are disjoint then

$$|\text{vert}(U_0 \cup \ldots \cup U_t)| = \sum_{j=0}^{t} |\text{vert}(U_j)| = \sum_{j=0}^{t} |U_j| \ .$$

[9] Since we consider the containers in reverse order, t is the "first" index satisfying this condition.

We thus conclude that $\sum_{j=0}^{t}|U_j| \leq \frac{(t+1)\varepsilon n}{4k}$. By Claim 3.3, the sequence Δ_t is decreasing with respect to t. Hence, in (4), the contribution of $|U_t|$ decreases as t increases, implying that the sum is maximized when $|U_t| = \frac{\varepsilon n}{4k}$ for every $0 \leq t \leq 4k/\varepsilon - 1$. We can thus bound (4) from above by

$$\frac{\varepsilon}{4}n^2 + \frac{\varepsilon n}{4k}\sum_{t=0}^{4k/\varepsilon-1}\Delta_t = \frac{\varepsilon}{4}n^2 + \frac{\varepsilon n}{4k}\Delta_0 + \frac{\varepsilon n}{4k}\sum_{t=1}^{4k/\varepsilon-1}\Delta_t \leq \frac{\varepsilon}{2}n^2 + \frac{\varepsilon n}{4k}\sum_{t=1}^{4k/\varepsilon-1}\Delta_t\ , \tag{5}$$

where in the last inequality we used the fact that $\Delta_0 \leq |\bar{V}| = kn$. By Claim 3.4, for every $t \geq 1$ we have $\Delta_t \leq |C_{t-1}(I)\backslash C_t(I)|$, implying that the RHS of (5) is at most

$$\frac{\varepsilon}{2}n^2 + \frac{\varepsilon n}{4k}\sum_{t=1}^{4k/\varepsilon-1}|C_{t-1}(I)\backslash C_t(I)| \leq \frac{\varepsilon}{2}n^2 + \frac{\varepsilon n}{4k}\cdot|C_0(I)\backslash C_{4k/\varepsilon-1}(I)| < \frac{\varepsilon}{2}n^2 + \frac{\varepsilon n}{4k}\cdot kn < \varepsilon n^2\ ,$$

thus completing the proof. ◀

We now extract the following useful corollary. We remind the reader of the notation S_ϕ in Fact 3.7. If S_ϕ is such a subset of $\bar{V}$ then we will use F_ϕ in order to name subsets of S_ϕ (we mention this since we could call such subsets F without the subscript ϕ).

▶ **Corollary 3.11.** *Given a graph $G = (V, E)$ and an integer k let $H_{G,k} = (\bar{V}, \bar{E})$ be the graph from Definition 3.6. Suppose that for some $\varepsilon > 0$ the graph G is ε-far from being k-colorable. Suppose $S = \{v_1, \ldots, v_s\}$ is such that $G[S]$ is k-colorable with coloring $\phi : S \mapsto [k]$. Then there is $1 \leq t \leq 4k/\varepsilon$ and an $F_\phi \subseteq S_\phi$ of size $|F_\phi| = t$ so that*

- **(i)** *$C_t(F_\phi) = C_t(S_\phi)$.*
- **(ii)** *$|vert(C_t(F_\phi))| \leq \left(1 - \frac{\varepsilon t}{4k}\right)n$.*

Here, C_t refers to the containers produced by Algorithm-1.

Proof. By Fact 3.7 we know that S_ϕ is an independent set in $H_{G,k}$. By Lemma 3.10 we know that there is a $1 \leq t \leq 4k/\varepsilon$ so that $|\mathrm{vert}(C_t(S_\phi))| \leq \left(1 - \frac{\varepsilon t}{4k}\right)n$. Let $F_t \subseteq S_\phi$ be the fingerprint of $C_t(S_\phi)$. Then by Claim 3.2 we have $C_t(F_t) = C_t(S_\phi)$ so we also have $|\mathrm{vert}(C_t(F_t))| \leq \left(1 - \frac{\varepsilon t}{4k}\right)n$. Finally note that $|F_t| = t$ (by Algorithm-1) and thus we can take F_t as the set F_ϕ in the statement. ◀

3.4 Proof of Theorem 1.2

Suppose $G = (V, E)$ is a graph on n vertices and G is ε-far from being k-colorable. Let $S \subset V$ be a random set of vertices of size $s = C\frac{k}{\varepsilon}\ln(k/\varepsilon)$, drawn uniformly at random and without replacements, where C is a large enough constant to be determined below. We need to show that the probability that $G[S]$ is k-colorable is at most $1/3$. Denote the vertices of S by $\{u_1, \ldots, u_s\}$ so that $[s]$ is the set of indices of these vertices. Let $H_{G,k} = (\bar{V}, \bar{E})$ be the graph from Definition 3.6. If I is an independent set in $H_{G,k}$ then we say that a container $C_t(I) \subset \bar{V}$ is *small* if $|\mathrm{vert}(C_t(I))| \leq \left(1 - \frac{\varepsilon t}{4k}\right)n$.

For every subset of indices $R \subseteq [s]$ let $R^* = \{u_i : i \in R\}$ (so $R^* \subseteq S$). If R is as above and $\phi : R \mapsto [k]$ then we let $R^*_\phi = \{u_{i,\phi(i)} : i \in R\}$ (so $R^*_\phi \subseteq \bar{V}$ as in Fact 3.7). For every subset of indices $R \subseteq [s]$ of size $t = |R| \leq \frac{4k}{\varepsilon}$ and every coloring function $\phi : R \mapsto [k]$, let $X_{R,\phi}$ be the event that R^*_ϕ is an independent set in $H_{G,k}$, that $C_t(R^*_\phi)$ is a small container, and that ϕ can be extended to coloring of S so that $C_t(R^*_\phi) = C_t(S_\phi)$, where we refer to containers of Algorithm-1 running on $H_{G,k}$. Let $X = \bigcup_{R,\phi} X_{R,\phi}$.

We claim that if $G[S]$ is k-colorable, then event X must hold. Indeed, if ϕ is a k-coloring of $G[S]$ and $F_\phi \subseteq S_\phi$ is the set satisfying Corollary 3.11 then F_ϕ determines the set R and ϕ of one of the events $X_{R,\phi}$ that must hold. This means that if we let C denote the event that $G[S]$ is k-colorable then $\mathbb{P}[C] \leq \sum_{X_{R,\phi}} \mathbb{P}[C|X_{R,\phi}]$. We will now complete the proof by showing that this last expression is at most $1/3$.

Consider one of the events $X_{R,\phi}$ with $t = |R|$ and assume we first sampled the vertices R^* and then the remaining $s - t$ vertices. If event $X_{R,\phi}$ holds then $|\text{vert}(C_t(R^*_\phi))| \leq \left(1 - \frac{\varepsilon t}{4k}\right) n$ and the remaining $s - t$ vertices of S should be such that ϕ can be extended to all of S so that $C_t(R^*_\phi) = C_t(S_\phi)$. But this means in particular that these $s - t$ vertices should all belong to $\text{vert}(C_t(R^*_\phi))$. Indeed, if $s_i \in S$ does not belong to $\text{vert}(C_t(R^*_\phi))$, then for every color q, the vertex $v_{i,q} \notin C_t(R^*_\phi)$ implying that we cannot extend ϕ from R^* to S so that $S_\phi \subseteq C_t(R^*_\phi)$. But by the basic properties of containers $S_\phi \subseteq C_t(S_\phi)$ so we cannot have $C_t(R^*_\phi) = C_t(S_\phi)$. We thus conclude that

$$\mathbb{P}[C|X_{R,\phi}] \leq \left(\frac{|\text{vert}(C_t(R^*_\phi))|}{n}\right)^{s-t} \leq \left(1 - \frac{\varepsilon t}{4k}\right)^{s-t} .$$

Applying the union bound over all $1 \leq t \leq 4k/\varepsilon$, all possible choices of subsets $R \subseteq [s]$ of size $|R| = t$, and all the possible colorings[10] ϕ, we have

$$\mathbb{P}[C] \leq \sum_{X_{R,\phi}} \mathbb{P}[C|X_{R,\phi}] \leq \sum_{t=1}^{4k/\varepsilon} \binom{s}{t} \cdot k^t \cdot \left(1 - \frac{\varepsilon t}{4k}\right)^{s-t} \leq \sum_{t=1}^{4k/\varepsilon} \exp\left(t \ln(ks) - \frac{\varepsilon t s}{8k}\right) ,$$

where in the last inequality we use the fact that $s \geq 2t$. It is easy to see that if $s = C\frac{k}{\varepsilon}\log\frac{k}{\varepsilon} = O(\frac{k}{\varepsilon}\log\frac{1}{\varepsilon})$, for[11] a constant C large enough, then the above expression is bounded from above by $1/3$. We conclude that the probability that $G[S]$ is colorable, given that G itself is ε-far from being k-colorable, is at most $1/3$, thus completing the proof.

References

1 N. Alon and M. Krivelevich. Testing k-colorability. *SIAM Journal on Discrete Mathematics*, 15:211–227, 2002. doi:10.1137/S0895480199358655.

2 Noga Alon and Asaf Shapira. Testing satisfiability. *Journal of Algorithms*, 47:87–103, 2003. doi:10.1016/S0196-6774(03)00019-1.

3 G. Andersson and L. Engebretsen. Property testers for dense constraint satisfaction programs on finite domains. *Random Structures and Algorithms*, 21:14–32, 2002. doi:10.1002/rsa.10041.

4 J. Balogh, R. Morris, and W. Samotij. Independent sets in hypergraphs. *Journal of the American Mathematical Society*, 28:669–709, 2015.

5 A. Bhattacharyya and Y. Yoshida. *Property Testing: Problems and Techniques*. Springer Nature, 2022.

6 E. Blais and C. Seth. New graph and hypergraph container lemmas with applications in property testing. In *Proc. of STOC 2024*, pages 1793–1804, 2024. doi:10.1145/3618260.3649708.

7 E. Blais and C. Seth. Testing graph properties with the container method. *SIAM Journal on Computing*, 54(6):1626–1642, 2025. doi:10.1137/24m1653173.

8 A. Bogdanov and F. Li. A better tester for bipartiteness? arXiv preprint. arXiv:1011.0531.

9 A. Bogdanov and L. Trevisan. Lower bounds for testing bipartiteness in dense graphs. In *Proc. of CCC 2004*, pages 75–81, 2004. doi:10.1109/CCC.2004.1313803.

[10] Note (crucially) that we only need to consider coloring of R not of all of S.

[11] The second equality relies on the observation we made on Page 1, that if G is ε-from being k-colorable then $k \leq 1/\varepsilon$.

10 B. Bollobás, P. Erdős, M. Simonovits, and E. Szemerédi. Extremal graphs without large forbidden subgraphs. *Annals of Discrete Mathematics*, 3:29–41, 1978.

11 A. Czumaj and C. Sohler. Testing hypergraph coloring. In *Proc. of ICALP 2001*, pages 493–505, 2001. doi:10.1007/3-540-48224-5_41.

12 P. Erdős. On circuits and subgraphs of chromatic graphs. *Mathematika*, 9:170–175, 1962.

13 N. Fiat and D. Ron. On efficient distance approximation for graph properties. In *Proc. of SODA 2021*, pages 1618–1637, 2021.

14 A. Frieze and R. Kannan. The regularity lemma and approximation schemes for dense problems. In *Proc. of FOCS 1996*, pages 12–20, 1996.

15 A. Frieze and R. Kannan. Quick approximation to matrices and applications. *Combinatorica*, 19:175–220, 1999. doi:10.1007/s004930050052.

16 O. Goldreich. *Introduction to property testing*. Cambridge University Press, 2017.

17 O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45:653–750, 1998. doi:10.1145/285055.285060.

18 O. Goldreich and D. Ron. Algorithmic aspects of property testing in the dense graphs model. *SIAM Journal on Computing*, 40:376–445, 2011. doi:10.1137/090749621.

19 O. Goldreich and L. Trevisan. Three theorems regarding testing graph properties. *Random Structures and Algorithms*, 23:23–57, 2003. doi:10.1002/rsa.10078.

20 O. Goldreich and A. Wigderson. Non-adaptive vs adaptive queries in the dense graph testing model. In *Proc. of FOCS 2021*, pages 269–275, 2021.

21 M. Gonen and D. Ron. On the benefits of adaptivity in property testing of dense graphs. *Algorithmica*, 58:811–830, 2010. doi:10.1007/s00453-008-9237-4.

22 D. Kleitman and K. Winston. On the number of graphs without 4-cycles. *Discrete Mathematics*, 41:167–172, 1982. doi:10.1016/0012-365X(82)90204-7.

23 T. Łuczak. The chromatic number of random graphs. *Combinatorica*, 11:45–54, 1991. doi:10.1007/BF01375472.

24 M. Parnas, D. Ron, and R. Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72:1012–1042, 2006. doi:10.1016/j.jcss.2006.03.002.

25 V. Rödl and R. Duke. On graphs with small subgraphs of large chromatic number. *Graphs and Combinatorics*, 1:91–96, 1985. doi:10.1007/BF02582932.

26 A. Sapozhenko. Systems of containers and enumeration problems. In *Stochastic Algorithms: Foundations and Applications (SAGA 2005)*, pages 1–13, 2005.

27 D. Saxton and A. Thomason. Hypergraph containers. *Inventiones mathematicae*, 201:925–992, 2015.

28 C. Sohler. Almost optimal canonical property testers for satisfiability. In *Proc. of FOCS 2012*, pages 541–550, 2012.

29 E. Szemerédi. Regular partitions of graphs. In J. C. Bermond, J. C. Fournier, M. Las Vergnas, and D. Sotteau, editors, *Proc. Colloque Inter. CNRS*, pages 399–401, 1978.

Distributed Gaussian Mean Testing Under Communication Constraints: Messages, Samples, and Coins

Clément L. Canonne
University of Sydney, Australia

Nimitt
IIT Gandhinagar, India

Abstract

We revisit the problem of Gaussian mean testing in a distributed, communication constrained setting, where each of n users independently observes samples from an unknown d-dimensional spherical Gaussian distribution $\mathcal{G}(\boldsymbol{\mu}, \mathbb{I}_d)$, and can communicate up to ℓ bits to a central referee. The referee's goal is then to distinguish between cases (i) $\|\boldsymbol{\mu}\|_2 = 0$ versus (ii) $\|\boldsymbol{\mu}\|_2 \geq \varepsilon$. This problem has been considered in the private- and public-coin settings, when each user holds exactly *one* sample, or more generally when each holds exactly m samples. In this work, we significantly generalize the question in three directions: when the users only share a small number s of random bits, when each user holds a different number of samples m_k, and when each user can send a different number of bits ℓ_k to the referee.

2012 ACM Subject Classification Theory of computation → Communication complexity; Theory of computation → Distributed algorithms; Theory of computation → Randomness, geometry and discrete structures

Keywords and phrases Distribution Testing, Property Testing, Distributed Algorithms, Communication Constraints

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.36

Category RANDOM

Related Version *Preprint*: `https://arxiv.org/abs/2605.29426`

Acknowledgements The authors thank the anonymous reviewers for their helpful suggestions.

1 Introduction

You have been put in charge of a new program, to monitor underwater volcanic activity in a large sector of the ocean. To maximize coverage, you have identified a set of n locations where to deploy buoys equipped with high-tech measuring devices, strategically chosen to maximize coverage. Each of these sensors will measure the volcanic activity at regular intervals, and send hourly the result to the central station for processing and analysis, to detect any risk. But choosing the location of these buoys was only the first step: the ocean is a harsh and big place, and as a result these sensors are not only quite distant from the station, they also have strong limitations on how many good-quality measurements they can make and how much data they can accurately transmit before running out of energy for the day. The measurements they make are subject to significant background noise; and even broadcasting data *to* the network of buoys to synchronize them is complex, given how remote they all are. So you are left with a daunting, highly constrained task:

> *How to design a good detection protocol under heterogeneous constraints, where each sensor can make a different number of measurements, send a different amount of data back to the center, and they all only share a limited amount of coordination?*

© Clément L. Canonne and Nimitt;
licensed under Creative Commons License CC-BY 4.0

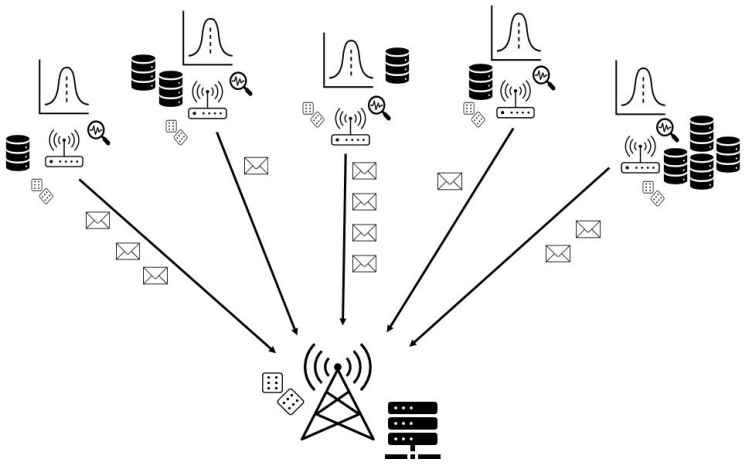


Figure 1 A depiction of the setting considered in this work: several distributed machines ("users") are measuring the same signal (vector $\boldsymbol{\mu} \in \mathbb{R}^d$) subject to white noise (spherical Gaussian). They all share a short random seed, and have different capabilities (amount of data collected, amount of communication they can send): by sending a message to a central server ("referee"), they must enable it to test whether the underlying signal measure exists ($\|\boldsymbol{\mu}\|_2 > \varepsilon$) or is pure noise ($\boldsymbol{\mu} = \mathbf{0}$).

Problem Statement

We formulate the problem as a composite hypothesis testing task. Let $\mathcal{G}(\boldsymbol{\mu}, \mathbb{I}_d)$ denote a d-dimensional Gaussian distribution with mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$ and covariance $\mathbb{I}_d$. Define the following hypothesis classes for a pre-specified threshold $\varepsilon \in (0, 1]$:

$$\mathcal{H}_0 := \{\mathcal{G}(\mathbf{0}, \mathbb{I}_d)\} \quad \mathcal{H}_1 := \{\mathcal{G}(\boldsymbol{\mu}, \mathbb{I}_d) : \|\boldsymbol{\mu}\|_2 \geq \varepsilon\}$$

In a distributed setting, n users observe i.i.d. samples from an unknown distribution $G \in \mathcal{H}_0 \cup \mathcal{H}_1$ and transmit ℓ-bit messages to a central referee. Specifically, user $k \in [n]$ observes $m_k \geq 1$ independent samples from G:

$$X^{(k)} = \left(\boldsymbol{X}^{(k,1)}, \boldsymbol{X}^{(k,2)}, \cdots, \boldsymbol{X}^{(k,m_k)}\right), \quad \boldsymbol{X}^{(k,i)} \sim G.$$

and applies a (possibly randomized) encoding function $W^{(k)} \colon \mathbb{R}^{d \cdot m_k} \to \mathcal{Y}_k := \{0,1\}^{\ell_k}$ to it, before sending the resulting message $\boldsymbol{Y}^{(k)} = W^{(k)}\left(X^{(k)}\right)$ to the referee. The referee then applies a test $T \colon \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_n \to \{0,1\}$ to the vector of received messages $Y^n = \left(\boldsymbol{Y}^{(1)}, \boldsymbol{Y}^{(2)}, \cdots, \boldsymbol{Y}^{(n)}\right)$, and accepts $\mathcal{H}_0$ if, and only if, $T(Y^n) = 0$.

We allow the users to share $s \geq 0$ bits of *public randomness*, which are used to jointly sample the tuple encoding functions $W^n = (W^{(1)}, W^{(2)}, \cdots, W^{(n)})$.[1] We define the *two-sided error* for protocol $\Pi := (W^n, T)$ as

$$\delta(\Pi) = \max\left(\sup_{G \in \mathcal{H}_0} \Pr\left[T(Y^n) = 1\right], \sup_{G \in \mathcal{H}_1} \Pr\left[T(Y^n) = 0\right]\right)$$

The goal is to find required amount of *resources* (amount of shared randomness s, numbers of samples $(m_k)_k$, individual communication budgets $(\ell_k)_k$) such that there exists a protocol Π with two-sided error bounded by a target error probability δ (unless specified otherwise, fixed as a small constant $1/10$). For simplicity, we decompose the problem into following three subproblems:

[1] That is, the encoding functions (or "channels") are randomized using both $s \geq 0$ bits of public randomness, and an arbitrary number of bits of private randomness (independent across users).

1. **Limited public coin**: Each user observes only one sample, and sends ℓ bits ($m_k = 1$, $\ell_k = \ell$ for all k); the users are allowed to share s bits of public randomness. What is the *sample complexity*, that is, the minimum $n = n(d, s, \ell, \varepsilon)$ such that $\inf_\Pi \delta(\Pi) \le 1/10$?
2. **Heterogeneous samples**: The users share $s = \Omega(\log(d/\ell))$ bits of public randomness, and each sends ℓ bits ($\ell_k = \ell$ for all k). What are sufficient conditions on $\boldsymbol{m} = (m_1, \dots, m_n)$ such that there exists a protocol Π with $\delta(\Pi) \le 1/10$?
3. **Heterogeneous communication**: The users share $s = \Omega(\log(d/\ell))$ bits of public randomness, and each observes one sample ($m_k = 1$ for all k). What are sufficient conditions on $\boldsymbol{\ell} = (\ell_1, \dots, \ell_n)$ such that there exists a protocol Π with $\delta(\Pi) \le 1/10$?

Our results

We now summarize our results for these three subproblems.

▶ **Theorem 1** (Limited public-coin). *For $\ell \in [d]$ and $s \ge 0$, there exists an s-public coin protocol for distributed Gaussian mean testing under ℓ-bit communication constraint with*

$$O\left(\frac{d}{\ell\varepsilon^2} \cdot \sqrt{\max\left(\frac{d}{2^{\Theta(s)}}, \ell\right)}\right)$$

users, each holding one sample.

Note that for $s = 0$ and $s = O(\log(d/\ell))$, this matches the *private-* and *public-coin* bounds of $O(d^{3/2}/(\ell\varepsilon^2))$ and $O(d/(\sqrt{\ell}\varepsilon^2))$ from [3], known to be tight [12].

▶ **Theorem 2** (Heterogeneous samples). *For $\ell \in [r, rd]$, $s = \Omega(\log(d/\ell))$, and $\boldsymbol{m} = (m_1, \dots, m_n) \in \mathbb{Z}_+^n$ satisfying $m_k \in [r, r\varepsilon^{-2}]$ for all $k \in [n]$, there exists a protocol for distributed Gaussian mean testing under ℓ-bit communication constraint with n users provided that*

$$\frac{\sum_{1 \le k_1 \ne k_2 \le n} \sqrt{m_{k_1} m_{k_2}}}{n} \ge C \cdot \frac{d}{\sqrt{\ell}\varepsilon^2}$$

where $C, r > 0$ are absolute constants.

To interpret this result, observe that, at one extreme, with $m_k = \Theta(1)$ for all k, the expression matches the *public-coin* bound. Moreover, we note that for a fixed "budget" of samples $m = \sum_{k=1}^n m_k$, a "balanced" allocation of samples across users maximizes the left-hand side, and hence is better (at least for our testing protocol).

▶ **Theorem 3** (Heterogeneous communication). *For $\boldsymbol{\ell} = (\ell_1, \dots, \ell_n) \in \mathbb{Z}_+^n$ and $s = \Omega(\log(d/\|\boldsymbol{\ell}\|_\infty))$, there exists a protocol for distributed Gaussian mean testing under $\boldsymbol{\ell}$-bit communication constraint with n users, where user k has one sample and can send ℓ_k bits to the referee, provided that*

$$\frac{\|\boldsymbol{\ell}\|_1}{\sqrt{\|\boldsymbol{\ell}\|_\infty}} \ge C \cdot \frac{d}{\varepsilon^2}$$

where $C > 0$ is an absolute constant.

Letting $\tilde{\ell} = \frac{1}{n}\sum_{k=1}^{n} \ell_k$ be the average communication budget, one can rephrase the sufficient condition as

$$n \cdot \frac{\tilde{\ell}}{\sqrt{\max_{1\le k\le n} \ell_k}} \ge C \cdot \frac{d}{\varepsilon^2}$$

which, for a fixed value of $\tilde{\ell}$, is easiest to satisfy when all communication constraints are the same ("balanced").

We emphasize that this separation in three subproblems is for clarity and simplicity of exposition: we also provide a protocol involving a "mix-and-match" of all parameters, where public randomness s, sample counts $\boldsymbol{m} = (m_1, \dots, m_n)$, and individual communication budgets $\boldsymbol{\ell} = (\ell_1, \dots, \ell_n)$ are all kept as free parameters. The guarantees of this protocol are summarized in the following theorem:

▶ **Theorem 4** (Everything, everywhere, all at once)**.** *For $s \ge 0$, $\boldsymbol{m} = (m_1, \dots, m_n) \in \mathbb{Z}_+^n$, and $\boldsymbol{\ell} = (\ell_1, \dots, \ell_n) \in \mathbb{Z}_+^n$, there exists a protocol for distributed Gaussian mean testing with n users sharing s bits of public randomness, where user k has m_k samples and can send ℓ_k bits to the referee, provided that there exists a partition $\mathcal{P} = \{P_1, \dots, P_K\}$ of the n users such that*

$$\frac{\sum_{j_1 \ne j_2 \in [K]} \sqrt{\min_{i_1 \in P_{j_1}}(m_{i_1}) \cdot \min_{i_2 \in P_{j_2}}(m_{i_2})}}{K} \ge C \cdot \frac{d}{\varepsilon^2 \sqrt{\max\left(\frac{d}{2^{\Theta(s)}}, \|\boldsymbol{\ell}\|_\infty\right)}}$$

and, for every $P \in \mathcal{P}$, $\sum_{i\in P} \ell_i \ge r \cdot \max\left(\frac{d}{2^{\Theta(s)}}, \|\boldsymbol{\ell}\|_\infty\right)$ and $\min_{i\in P}(m_i) \in [r, r\varepsilon^{-2}]$; where $C, r > 0$ are absolute constants.

Related work

In the centralized setting where all samples are fully available to the referee, the tight sample complexity of for Gaussian mean testing is known, and, while considered "folklore", can for instance be found in [7, 10] (as a special case of more general problem). Subsequent work has expanded on this foundation, exploring Gaussian mean testing under communication constraints [3] and truncation models [9]. [3] studied the problem in two variants of homogeneous regime (one sample and same communication budget ℓ for all users), namely the *private-coin* ($s = 0$) and *public-coin* ($s = \infty$), and showed sample complexities of

$$O\left(\frac{d^{3/2}}{\ell\varepsilon^2}\right) \quad (\textit{private coin}), \; O\left(\frac{d}{\sqrt{\ell}\varepsilon^2}\right) \quad (\textit{public coin})$$

While the authors only established upper bounds and (limited) lower bounds, follow-up work [11, 12] showed these two sample complexities to be tight. (Moreover, exploiting standard properties of the Gaussian distribution, the above easily generalizes to the case where each of the n users holds the same number of samples $m \ge 1$.[2])

The above works reveal a $\sqrt{d/\ell}$ gap between the private- and public-coin sample complexities, highlighting the value of shared randomness. However, the public-coin protocol of [3] (as well as the similar protocol of [12]) rely on jointly sampling a uniformly random rotation

[2] Namely, if $\boldsymbol{X}^{(k,1)}, \boldsymbol{X}^{(k,2)}, \cdots, \boldsymbol{X}^{(k,m_k)} \sim \mathcal{G}(\boldsymbol{\mu}, \mathbb{I}_d)$, then $\tilde{\boldsymbol{X}}^{(k)} := \frac{1}{\sqrt{m}}\sum_{i=1}^{m} \boldsymbol{X}^{(k,i)} \sim \mathcal{G}(\sqrt{m}\boldsymbol{\mu}, \mathbb{I}_d)$, so one can use the one-sample protocol with new distance parameter $\varepsilon' = \sqrt{m}\varepsilon$ and "single sample per user" $\tilde{\boldsymbol{X}}^{(k)}$.

of the space, which technically requires infinitely many shared random bits – even with adequate discretization, the amount of shared randomness necessary is still impractical. One can thus wonder what can be achieved with a *smaller* "shared randomness budget:" similarly, what happens if each user holds a different number of samples m_k, or can communicate a different number of bits ℓ_k. To the best of our knowledge, all of these remain essentially understudied and open in the context of Gaussian mean testing, and even distribution testing at large, with the two exceptions outlined below.

On the "limited randomness" side, our techniques and results for the s-public coin setting are reminiscent of the work of [1], which study the problem of *identity testing for discrete distributions* under communication constraints. They provide an optimal protocol for identity testing of distributions over a finite domain $[k]$ under ℓ-bit communication constraint with

$$n = \Theta\left(\frac{\sqrt{k}}{\varepsilon^2}\sqrt{\frac{d}{2^\ell}\vee 1}\sqrt{\frac{k}{2^{\ell+s}}\vee 1}\right)$$

users, giving clear interpolation between *private-* and *public-coin*. Our Blockwise Randomized Hadamard Transform (BRHT) primitive (see Section 3) for compression is similar to their technique "Domain Compression", showing an exponential saving (in the number of shared random bits) in the sample complexity of the problem. Interestingly, however, both the actual techniques used, and the resulting tradeoffs, are quite incomparable: the upshot of [1] was that "one bit of communication is equivalent to two bits of shared randomness" (in terms of sample complexity), which does no longer hold in the Gaussian mean testing setting.

On the "heterogeneous communication" side, while we are not aware of previous work for the case of Gaussian mean testing, we note that in the case of private-coin protocols for testing of discrete distributions, a baseline upper bound is hinted at in [6, Exercise 4.4], while *lower bounds* for *estimation* (not testing) are derived in [2].

We conclude by mentioning that, in the (as discussed above, analogous, yet orthogonal) setting of identity testing of discrete distributions, under local privacy constraints, the recent work [8] states that *"it is not at all obvious how this [heterogeneous sample sizes] could be handled neatly, and general results for this model could greatly help practical implementations."* While our results do not directly address their question (as we deal with mean testing and not discrete distributions, and communication instead of local privacy constraints), we hope that some of our ideas can help in this regard, especially given the observed similarities between communication and local privacy constraints [2].

2 Preliminaries

Notation

Throughout, we write $[n] := \{1, 2, \ldots, n\}$, and denote by $\boldsymbol{u}_d := (1/2, 1/2, \ldots, 1/2) \in \mathbb{R}^d$ the mean vector of a d-dimensional binary product uniform distribution over $\{0,1\}^d$, and by $\mathbb{I}_d$ the d-dimensional identity matrix. We use the standard asymptotic notations $O(\cdot), \Omega(\cdot)$ and $\Theta(\cdot)$.

Simplification

For clarity of exposition, we assume throughout that $d = 2^{k_1}$ and $\ell = 2^{k_2}$ for some integers $k_1, k_2 \geq 0$. This assumption is without loss of generality up to constant factors. Indeed, for arbitrary d, one can embed into dimension d' with

$$d \leq d' < 2d, \qquad d' = 2^{k_1}$$

via zero-padding. Similarly, for arbitrary ℓ, one can use ℓ' bits with

$$\ell/2 < \ell' \leq \ell, \qquad \ell' = 2^{k_2}.$$

We will heavily rely on the following tools and techniques from the literature, which we recall below.

▶ **Definition 5** (Normalized Sylvester Hadamard Matrix). *The* normalized Sylvester Hadamard matrix $H_d \in \mathbb{R}^{d\times d}$ *is recursively defined by*

$$H_1 := [1],$$

and for every integer $d \geq 1$

$$H_{2d} := \frac{1}{\sqrt{2}}\begin{bmatrix} H_d & H_d \\ H_d & -H_d \end{bmatrix}$$

▶ **Lemma 6** (Paley–Zygmund inequality). *For a non-negative random variable* Z *and* $0 \leq \theta \leq 1$

$$\Pr[Z > \theta\mathbb{E}[Z]] \geq (1-\theta)^2\frac{\mathbb{E}[Z]^2}{\mathbb{E}[Z^2]}$$

Binary product mean testing

Let P be an unknown d-dimensional binary product distribution over $\{0,1\}^d$ with mean vector $\boldsymbol{p} \in (0,1)^d$. Given i.i.d. samples from P and a threshold $\varepsilon > 0$, the binary product mean testing problem is to distinguish between cases $\boldsymbol{p} = \boldsymbol{u}_d$ versus $\|\boldsymbol{p} - \boldsymbol{u}_d\|_2 \geq \varepsilon$ with probability at least 99/100. It is known that the optimal sample complexity for this problem is $\Theta(\sqrt{d}/\varepsilon^2)$, and can be achieved by Algorithm 1 below. For completeness, we provide the analysis of Algorithm 1 in Appendix, Theorem 21.

Algorithm 1 Binary product mean testing.

Require: Samples $\boldsymbol{X}^{(1)}, \ldots, \boldsymbol{X}^{(n)} \in \{0,1\}^d$; threshold $\varepsilon > 0$.

$\tau \leftarrow \varepsilon^2/2$
for $i \in [d]$ **do**
  $T_i \leftarrow \sum_{k_1 \neq k_2 \in [n]} (X_i^{(k_1)} - 1/2)(X_i^{(k_2)} - 1/2)$
$T \leftarrow \frac{1}{n(n-1)} \sum_{i=1}^d T_i$
if $T > \tau$ **then**
  return reject
else
  return accept

Private-coin

As discussed above, the private-coin setting, which corresponds to $s = 0$ bits of shared randomness, was studied for Gaussian mean testing in [3]. We briefly describe their protocol, as we will use it later on, which uses two main ingredients: a reduction to the binary product case, and a simple private-coin algorithm to solve the latter.

▶ **Lemma 7** (Reduction to binary product mean testing). *Let* $\varepsilon \in (0,1)$. *If there exists an algorithm for a binary product mean testing with* $n(d,\varepsilon)$ *samples, there exists an algorithm for Gaussian mean testing with* $n(d, \varepsilon/\sqrt{8})$.

Proof. Let $\boldsymbol{X}^{(1)}, \dots, \boldsymbol{X}^{(n)}$ be i.i.d. samples from $\mathcal{G}(\boldsymbol{\mu}, \mathbb{I}_d)$. Define

$$\boldsymbol{Y}^{(k)} := \mathbb{I}(\boldsymbol{X}^{(k)} > 0) \in \{0,1\}^d.$$

Then the $\boldsymbol{Y}^{(k)}$ are i.i.d. samples from a binary product distribution P over $\{0,1\}^d$ with mean vector $\boldsymbol{p} \in (0,1)^d$. We distinguish the two cases,

Case 1: $\|\boldsymbol{\mu}\|_2 = 0$. Then, for all i, $p_i = \Pr[\boldsymbol{Y}_i^{(k)} = 1] = 1/2$. Thus, $\boldsymbol{p} = \boldsymbol{u}_d$.

Case 2: if $\|\boldsymbol{\mu}\|_2 \geq \varepsilon$. Then, for all i, $p_i = \Pr[\boldsymbol{Y}_i^{(k)} = 1] = \frac{1}{2}\operatorname{Erfc}\left(-\mu_i/\sqrt{2}\right)$. Using standard properties of Erfc, we have

$$\sum_{i=1}^{d}(p_i - 1/2)^2 \geq \frac{1}{8}\min\left(\sum_{i=1}^{d}\mu_i^2, d/2\right) = \frac{1}{8}\min\left(\|\boldsymbol{\mu}\|_2^2, d/2\right) \geq \frac{\varepsilon^2}{8}$$

Thus, $\|\boldsymbol{p} - \boldsymbol{u}_d\|_2 \geq \varepsilon/\sqrt{8}$.

Hence, to distinguish between the two cases, it is sufficient to test if $\boldsymbol{p} = \boldsymbol{u}_d$ versus $\|\boldsymbol{p} - \boldsymbol{u}_d\|_2 \geq \varepsilon/\sqrt{8}$, which is exactly the binary product mean testing problem. ◀

▶ **Theorem 8** ([3] Theorem 1). *For all $\ell \in [d]$, there exists a private-coin protocol for distributed Gaussian mean testing under ℓ-bit communication constraint with $O(d^{3/2}/\ell\varepsilon^2)$ users.*

Proof. Let $\boldsymbol{X}^{(1)}, \dots, \boldsymbol{X}^{(n)}$ be i.i.d. samples from $\mathcal{G}(\boldsymbol{\mu}, \mathbb{I}_d)$ where user k observes $\boldsymbol{X}^{(k)}$. Define $\boldsymbol{Y}^{(k)} := \mathbf{1}(\boldsymbol{X}^{(k)} > 0) \in \{0,1\}^d$. Then $\boldsymbol{Y}^{(k)}$ are i.i.d. samples from a binary product distribution P over $\{0,1\}^d$ with mean vector $\boldsymbol{p} \in (0,1)^d$. First, note that if $\ell = d$, each user can transmit the entire vector $\boldsymbol{Y}^{(k)}$, yielding a protocol with $\Theta(\sqrt{d}/\varepsilon^2)$ users using Lemma 7.

When $\ell < d$, we use the "*simulate and infer*" technique of [4]: d/ℓ users can simulate *one* sample of the d-dimensional product distribution, by having send (a different set of) ℓ coordinates. Combining this with the $\Theta(\sqrt{d}/\varepsilon^2)$ centralized sample complexity of binary product mean testing, this gives a protocol with sample complexity

$$\frac{d}{\ell} \cdot O\left(\frac{\sqrt{d}}{\varepsilon^2}\right) = O\left(\frac{d^{3/2}}{\ell\varepsilon^2}\right)$$

as claimed. ◀

3 Blockwise Randomized Hadamard Transform

In this section, we define and analyze the properties of a key primitive used throughout the paper, the *Blockwise Randomized Hadamard Transform (BRHT)*. At a high level, the idea is to replace a uniformly random rotation of $\mathbb{R}^d$ by a discrete analogue, which can be sampled using a *finite* number of random bits. An obvious candidate would be to use either a matrix of uniformly random ± 1; however, our aim is to preserve the ℓ_2 geometry, and this fails to do so. Indeed, we want an isometry, something which would play the same role as a uniformly random rotation, and in particular a transformation R which, when applied to a spherical Gaussian, would still result in a spherical Gaussian: which requires $R^\top R = \mathbb{I}_d$. This makes using a Hadamard matrix a very natural idea: however, with such a deterministic construction, we lack the randomness which enables more efficient testing protocols. To remedy this while still using a minimum amount of randomness, we consider a structured random orthogonal transform derived from the Hadamard matrix, which we define Blockwise Randomized Hadamard Transform (BRHT), and provides strong sketching guarantees with control over randomness requirement.

We note that structural randomized Hadamard transforms have previously been studied in areas such as fast dimensionality reduction and randomized linear algebra (e.g., the Fast Johnoson–Lindenstrauss Transform [5]). Our own primitive, BRHT, is tailored to the Gaussian mean testing setting, where the objective is not Johnson–Lindenstrauss embeddings but rather finite-randomness compression satisfying the block-wise energy preservation property established in Lemma 11. We formally define the BRHT below.

▶ **Definition 9** (Blockwise Randomized Hadamard Transform (BRHT))**.** *Let $d, L \in \mathbb{N}$ satisfy $L \mid d$, and define $b := d/L$. A (d, L)-*Blockwise Randomized Hadamard Transform (BRHT)* is the random matrix*

$$R := H_d D$$

where $H_d \in \mathbb{R}^{d\times d}$ is a normalized Sylvester Hadamard matrix and $D \in \mathbb{R}^{d\times d}$ is a blockwise diagonal matrix:

$$\mathrm{diag}(D) = \left(\boldsymbol{\delta}^{(1)}, \boldsymbol{\delta}^{(2)}, \cdots, \boldsymbol{\delta}^{(b)}\right)$$

where for every $i \in [b]$, $\boldsymbol{\delta}^{(i)} = (\delta_i, \delta_i, \cdots, \delta_i) \in \{-1, 1\}^L$ and $\delta_1, \ldots, \delta_b$ are 4-wise independent Rademacher random variables.

▶ **Lemma 10** (Orthogonality)**.** *Let R be a (d, L)-BRHT. Then $RR^\top = \mathbb{I}_d$.*

Proof. Since H_d is orthogonal and D is diagonal with entries in $\{\pm 1\}$, $RR^\top = (H_dD)(H_dD)^\top = H_dDD^\top H_d^\top = H_dH_d^\top = \mathbb{I}_d$. ◀

With this in hand, we are ready to state and prove the key property of BRHT our algorithms will hinge upon.

▶ **Lemma 11** ((d, L)-compression)**.** *Let R be a (d, L)-BRHT. Then, for every $\boldsymbol{\mu} \in \mathbb{R}^d$ and every $t \geq 1$,*

$$\Pr\left[\left\|(R\boldsymbol{\mu})_{[1:tL]}\right\|_2^2 \geq \frac{1}{100} \cdot \frac{tL}{d} \cdot \|\boldsymbol{\mu}\|_2^2\right] \geq \frac{8}{25},$$

where the probability is over the choice of R.

Proof. Let $b := d/L$ and $\rho := \|\boldsymbol{\mu}\|_2$. By Definition 9,

$$R = H_dD.$$

Define

$$Z := \left\|(R\boldsymbol{\mu})_{[1:tL]}\right\|_2^2.$$

Partition

$$\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_b), \qquad \boldsymbol{\mu}_i \in \mathbb{R}^L.$$

Using the Kronecker decomposition

$$H_d = H_b \otimes H_L,$$

the r-th block of $R\boldsymbol{\mu}$ satisfies

$$(R\boldsymbol{\mu})_{[(r-1)L+1:rL]} = \sum_{i=1}^{b} (H_b)_{r,i}\delta_i H_L \boldsymbol{\mu}_i.$$

Define

$$Z_r := \left\|(R\boldsymbol{\mu})_{[(r-1)L+1:rL]}\right\|_2^2,$$

so that

$$Z = \sum_{r=1}^{t} Z_r.$$

We first compute $\mathbb{E}[Z]$. Expanding Z_r gives

$$\begin{aligned} Z_r &= \left\| \sum_{i=1}^{b} (H_b)_{r,i}\delta_i H_L \boldsymbol{\mu}_i \right\|_2^2 \\ &= \sum_{i_1,i_2\in[b]} (H_b)_{r,i_1}(H_b)_{r,i_2}\delta_{i_1}\delta_{i_2} \left\langle H_L\boldsymbol{\mu}_{i_1}, H_L\boldsymbol{\mu}_{i_2}\right\rangle. \end{aligned}$$

Since H_L is orthogonal,

$$\langle H_L\boldsymbol{\mu}_{i_1}, H_L\boldsymbol{\mu}_{i_2}\rangle = \langle \boldsymbol{\mu}_{i_1}, \boldsymbol{\mu}_{i_2}\rangle.$$

Therefore,

$$Z_r = \sum_{i=1}^{b} (H_b)_{r,i}^2 \|\boldsymbol{\mu}_i\|_2^2 + \sum_{i_1\neq i_2} (H_b)_{r,i_1}(H_b)_{r,i_2}\delta_{i_1}\delta_{i_2}\langle \boldsymbol{\mu}_{i_1}, \boldsymbol{\mu}_{i_2}\rangle.$$

Since $(H_b)_{r,i}^2 = 1/b$,

$$Z_r = \frac{\rho^2}{b} + \sum_{i_1\neq i_2} (H_b)_{r,i_1}(H_b)_{r,i_2}\delta_{i_1}\delta_{i_2}\langle \boldsymbol{\mu}_{i_1}, \boldsymbol{\mu}_{i_2}\rangle.$$

By pairwise independence and symmetry of the Rademacher variables,

$$\mathbb{E}[\delta_{i_1}\delta_{i_2}] = 0 \qquad (i_1 \neq i_2),$$

and hence

$$\mathbb{E}[Z_r] = \frac{\rho^2}{b}.$$

Summing over $r \in [t]$,

$$\mathbb{E}[Z] = \frac{t\rho^2}{b}.$$

We now bound $\mathbb{E}[Z^2]$. For $r_1, r_2 \in [t]$,

$$\mathbb{E}[Z_{r_1}Z_{r_2}] = \frac{\rho^4}{b^2} + \sum_{\substack{i_1\neq i_2\\ j_1\neq j_2}} (H_b)_{r_1,i_1}(H_b)_{r_1,i_2}(H_b)_{r_2,j_1}(H_b)_{r_2,j_2} \cdot \mathbb{E}[\delta_{i_1}\delta_{i_2}\delta_{j_1}\delta_{j_2}]\langle \boldsymbol{\mu}_{i_1}, \boldsymbol{\mu}_{i_2}\rangle\langle \boldsymbol{\mu}_{j_1}, \boldsymbol{\mu}_{j_2}\rangle.$$

By 4-wise independence and symmetry of the Rademacher variables, only terms in which every δ_i appears with even multiplicity contribute to the expectation. Thus,

$$\mathbb{E}[Z_{r_1}Z_{r_2}] = \frac{\rho^4}{b^2} + 2\sum_{i_1\neq i_2} (H_b)_{r_1,i_1}(H_b)_{r_1,i_2}(H_b)_{r_2,i_1}(H_b)_{r_2,i_2}\langle \boldsymbol{\mu}_{i_1}, \boldsymbol{\mu}_{i_2}\rangle^2.$$

Using

$$(H_b)_{r,i} = \pm\frac{1}{\sqrt{b}}$$

and Cauchy–Schwarz,

$$\begin{aligned}\mathbb{E}[Z_{r_1}Z_{r_2}] &\le \frac{\rho^4}{b^2} + \frac{2}{b^2}\sum_{i_1\neq i_2}\|\boldsymbol{\mu}_{i_1}\|_2^2\|\boldsymbol{\mu}_{i_2}\|_2^2 \\ &\le \frac{3\rho^4}{b^2},\end{aligned}$$

where we used

$$\sum_{i=1}^{b}\|\boldsymbol{\mu}_i\|_2^2 = \rho^2.$$

Summing over $r_1, r_2 \in [t]$ yields

$$\mathbb{E}[Z^2] \le \frac{3t^2\rho^4}{b^2}.$$

Applying the Paley–Zygmund (Lemma 6),

$$\Pr[Z > \theta\mathbb{E}[Z]] \ge (1-\theta)^2\frac{\mathbb{E}[Z]^2}{\mathbb{E}[Z^2]} \ge \frac{(1-\theta)^2}{3}.$$

Choosing $\theta = 1/100$ gives

$$\Pr\left[Z > \frac{t\rho^2}{100b}\right] \ge \frac{8}{25}.$$

Since

$$b = \frac{d}{L},$$

the claim follows. ◀

The compression guarantees established in Lemma 11 are the key property of BRHT used throughout the remainder of the paper. Roughly speaking, this guarantees that, with constant probability, the retained block preserves $\Omega(L/d)$ fraction of the signal energy, allowing the original Gaussian mean testing problem to be compressed while retaining sufficient signal for testing. To conclude, we analyze the amount of randomness required to sample a BRHT.

▶ Remark 12. A (d, L)-BRHT requires at most $4\log(d/L)$ bits of randomness as (d/L) 4-wise Rademacher random variables can be generated using a standard construction involving a degree-3 polynomial over an appropriate field (see, e.g., [13, Corollary 3.34]).

▶ Remark 13 (Computational complexity). For $\boldsymbol{v} \in \mathbb{R}^d$, $H_d\boldsymbol{v}$ can be computed in $O(d\log d)$ time via the Fast Walsh–Hadamard transform. This computation dominates the running time of our protocols.

4 Limited randomness

In this section, we prove Theorem 1, by providing and analyzing an explicit protocol for the problem. This protocol is quite simple: it first uses the shared randomness to jointly sample a BRHT R with suitable parameters; each user then applies R to their samples to obtain an instance of Gaussian mean testing in a lower-dimensional space. Having used all the shared randomness available, they then run the private-coin protocol of Theorem 8 on this lower-dimensional problem.

▶ **Theorem 14** (Theorem 1, restated). *For $\ell \in [d]$ and $s \geq 0$, there exists an s-public coin protocol for distributed Gaussian mean testing under ℓ-bit communication constraint with*

$$O\left(\frac{d}{\ell\varepsilon^2} \cdot \sqrt{\max\left(\frac{d}{2^{\Theta(s)}}, \ell\right)}\right)$$

users, each holding one sample.

Proof. The users use $s/7$ bits of shared randomness to sample a (d, d_s)-BRHT R, where

$$d_s := d/2^{\lfloor s/28 \rfloor}\,.$$

This is possible by Remark 12, as sampling R requires at most $4\log(d/d_s)$ random bits. (We will explain the reason for the factor 7 in $s/7$ momentarily.)

Let $\boldsymbol{\mu} \in \mathbb{R}^d$ be an unknown mean vector and for all $k \in [n]$, $\boldsymbol{X}^{(k)} \sim \mathcal{G}(\boldsymbol{\mu}, \mathbb{I}_d)$ be the sample observed by user k. Define $L := d_s \vee \ell$ and $\tilde{\boldsymbol{\mu}} = (R\boldsymbol{\mu})_{[1:L]}$. Each user applies R to its sample to get $\tilde{\boldsymbol{X}}^{(k)} := (R\boldsymbol{X}^{(k)})_{[1:L]}$, so that $\tilde{\boldsymbol{X}}^{(k)} \sim \mathcal{G}\left(\tilde{\boldsymbol{\mu}}, \mathbb{I}_L\right)$. By Lemma 11, with probability at least $8/25$,

$$\|\tilde{\boldsymbol{\mu}}\|_2^2 \geq \frac{1}{100} \cdot \frac{L}{d} \cdot \|\boldsymbol{\mu}\|_2^2$$

Thus, if $\boldsymbol{\mu} = \mathbf{0}$, $\tilde{\boldsymbol{\mu}} = \mathbf{0}$ and otherwise $\|\boldsymbol{\mu}\|_2 \geq \varepsilon$ and with probability at least $8/25$, $\|\tilde{\boldsymbol{\mu}}\|_2^2 \geq \frac{L\varepsilon^2}{100d}$. The users and referee then use the private-coin protocol of Theorem 8 with $\tilde{\boldsymbol{X}}^{(k)}$ to distinguish between the two cases with

$$O\left(\frac{(L)^{3/2}}{\ell\left(\frac{L}{d}\varepsilon^2\right)}\right) = O\left(\frac{d\sqrt{L}}{\ell\varepsilon^2}\right)$$

users, (almost) proving the theorem. It remains to amplify the success probability.

Thus, with $O\left(\frac{d\sqrt{L}}{\ell\varepsilon^2}\right)$ users, if $\boldsymbol{\mu} = \mathbf{0}$, the referee accepts with probability at least $99/100$ (which is the failure probability from Theorem 8). But when $\|\boldsymbol{\mu}\|_2 \geq \varepsilon$, all we can say is that, by a union bound, the referee rejects with probability at least $1 - (\frac{17}{25} + \frac{8}{25} \cdot \frac{1}{100}) > 3/10$, as now the failure probability comes both from the choice of R and that of Theorem 8. Repeating the protocol 7 independent times, each time with new R, and accepting only if all repetitions output accept reduces the error probability below $1/10$. This increases the number of users only by a constant factor. Since $L = \max(d/2^{\lfloor s/28 \rfloor}, \ell)$, the theorem follows. ◀

With $s = O(\log(d/\ell))$, Theorem 1 yields sample complexity $O(d/\sqrt{\ell}\varepsilon^2)$, which matches the public-coin sample complexity in [3] when users must share an infinite amount of randomness.

▶ **Corollary 15** (Randomness-efficient optimal public-coin protocol). *There exists a protocol for distributed Gaussian mean testing under ℓ-bit communication constraints with $O(d/(\sqrt{\ell}\varepsilon^2))$ users, each holding one sample, and sharing $O(\log(d/\ell))$ bits of randomness.*

5 Heterogeneous samples

In this section, we prove Theorem 2. The core idea, which we refer to as *Signal Aggregation*, is to generalize the "naive strategy" previously discussed (see Footnote 2) where all users have the same number m of samples and aggregate them into a single "aggregated sample":

$$\boldsymbol{X}^{(k,1)}, \boldsymbol{X}^{(k,2)}, \cdots, \boldsymbol{X}^{(k,m_k)} \sim \mathcal{G}(\boldsymbol{\mu}, \mathbb{I}_d) \quad \rightsquigarrow \quad \tilde{\boldsymbol{X}}^{(k)} := \frac{1}{\sqrt{m}} \sum_{i=1}^{m} \boldsymbol{X}^{(k,i)} \sim \mathcal{G}(\sqrt{m}\boldsymbol{\mu}, \mathbb{I}_d)$$

Doing so effectively amplifies the signal, which means that one can then use a protocol designed for *one* sample per user, but with better distance parameter $\varepsilon' := \sqrt{m}\varepsilon$.

In the heterogeneous sample case, one can attempt to do the same thing, but now (since user k has m_k samples, while user k' has a possible different number of samples $m_{k'}$) the resulting aggregated samples no longer have the same distribution in the no-case (when $\boldsymbol{\mu} \neq \mathbf{0}$). Dealing with the fact that samples are no longer i.i.d. is the key technical difficulty in proving Theorem 2.

▶ **Theorem 16** (Theorem 2, restated). *For $\ell \in [r, rd]$, $s = \Omega(\log(d/\ell))$, and $\boldsymbol{m} = (m_1, \ldots, m_n) \in \mathbb{Z}_+^n$ satisfying $m_k \in [r, r\varepsilon^{-2}]$ for all $k \in [n]$, there exists a protocol (Algorithm 2) for distributed Gaussian mean testing under ℓ-bit communication constraint with n users provided that*

$$\frac{\sum_{1 \leq k_1 \neq k_2 \leq n} \sqrt{m_{k_1} m_{k_2}}}{n} \geq C \cdot \frac{d}{\sqrt{\ell}\varepsilon^2}$$

where $C, r > 0$ are absolute constants.

Proof. We first describe the protocol in Algorithm 2 and then analyze its correctness.

Algorithm 2 ℓ-bit, $\boldsymbol{m}$-sample distributed Gaussian mean testing.

User k, for every $k \in [n]$:

Require: $\forall j \in [m_k]$, $\boldsymbol{X}^{(k,j)} \sim \mathcal{G}(\boldsymbol{\mu}, \mathbb{I}_d)$; communication budget $\ell \in [7, 7d]$; $\forall t \in [7]$, R_t: $(d, 7d/\ell)$-BRHT

for $t \in [7]$ **do**
  Samples $\boldsymbol{X}^{(k,t)} \leftarrow \frac{1}{\sqrt{\lfloor m_k/7 \rfloor}} \sum_{j=(t-1)\lfloor m_k/7 \rfloor + 1}^{t\lfloor m_k/7 \rfloor} \boldsymbol{X}^{(k,j)}$
  $\tilde{\boldsymbol{X}}^{(k,t)} \leftarrow R_t \boldsymbol{X}^{(k,t)}$
  $\boldsymbol{y}^{(k,t)} \leftarrow \mathbf{1}\left(\tilde{\boldsymbol{X}}^{(k,t)}_{[1:\lfloor \ell/7 \rfloor]} > 0\right)$
Send $\boldsymbol{Y}^{(k)} = (\boldsymbol{y}^{(k,1)}, \ldots, \boldsymbol{y}^{(k,7)})$

Referee:

Require: Messages $\forall k \in [n]$, $\boldsymbol{Y}^{(k)} \in \{0,1\}^\ell$, threshold ε

$N \leftarrow \sum_{k_1 \neq k_2} \sqrt{\lfloor m_{k_1}/7 \rfloor \lfloor m_{k_2}/7 \rfloor}$
for $t \in [7]$ **do**
  Run Algorithm 1 on samples $\boldsymbol{y}^{(1,t)}, \ldots, \boldsymbol{y}^{(n,t)}$ with
    threshold $\varepsilon' = \frac{\varepsilon}{80}\sqrt{\frac{\ell N}{7dn(n-1)}}$ ▷ $\tau \leftarrow (\varepsilon')^2/2$
  if output is reject **then**
    return reject
return accept

The protocol performs 7 independent repetitions for probability amplification. We analyze a single repetition and omit the superscript (t) throughout. For simplicity, we also ignore flooring operations, as they affect the bounds only up to constant factors. Fix one repetition, and let R denote the corresponding $(d, 7d/\ell)$-BRHT. Define

$$\tilde{\boldsymbol{\mu}} := (R\boldsymbol{\mu})_{[1:\ell/7]}.$$

For every user k,

$$\boldsymbol{X}^{(k)} = \frac{1}{\sqrt{m_k/7}} \sum_{j=1}^{m_k/7} \boldsymbol{X}^{(k,j)} \sim \mathcal{G}\left(\sqrt{m_k/7}\,\boldsymbol{\mu}, \mathbb{I}_d\right).$$

Since R is orthogonal,

$$\tilde{\boldsymbol{X}}^{(k)} := (R\boldsymbol{X}^{(k)})_{[1:\ell/7]} \sim \mathcal{G}\left(\sqrt{m_k/7}\,\tilde{\boldsymbol{\mu}}, \mathbb{I}_{\ell/7}\right).$$

Define

$$\boldsymbol{y}^{(k)} = \mathbf{1}(\tilde{\boldsymbol{X}}^{(k)} > 0) \in \{0,1\}^{\ell/7}.$$

The vectors $\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(n)}$ are independent, though generally not identically distributed. For every coordinate $i \in [\ell/7]$, define

$$p_i^{(k)} := \Pr[y_i^{(k)} = 1], \qquad \tilde{p}_i^{(k)} := p_i^{(k)} - \frac{1}{2}.$$

The referee computes

$$T_i := \sum_{k_1 \neq k_2} (y_i^{(k_1)} - 1/2)(y_i^{(k_2)} - 1/2).$$

Since for all $k_1 \neq k_2$, $\tilde{p}_i^{(k_1)} \tilde{p}_i^{(k_2)} \geq 0$, applying Lemma 20 coordinate-wise yields

$$\mathbb{E}[T_i] = \sum_{k_1 \neq k_2} \tilde{p}_i^{(k_1)} \tilde{p}_i^{(k_2)},$$

and

$$\mathrm{Var}(T_i) \leq \frac{n(n-1)}{8} + (n-2)\mathbb{E}[T_i].$$

Define

$$T := \frac{1}{n(n-1)} \sum_{i=1}^{\ell/7} T_i.$$

Then

$$\begin{aligned}
\mathbb{E}[T] &= \frac{1}{n(n-1)} \sum_{i=1}^{\ell/7} \sum_{k_1 \neq k_2} \tilde{p}_i^{(k_1)} \tilde{p}_i^{(k_2)}, \\
\mathrm{Var}(T) &= \frac{1}{n^2(n-1)^2} \sum_{i=1}^{\ell/7} \mathrm{Var}(T_i) \\
&\leq \frac{\ell}{56\, n(n-1)} + \frac{(n-2)\mathbb{E}[T]}{n(n-1)}.
\end{aligned}$$

We now analyze the two cases:

Case 1: $\mu = 0$. In this case, $p_i^{(k)} = 1/2$ for all i, k, and hence $\tilde{p}_i^{(k)} = 0$. Therefore,

$$\mathbb{E}[T] = 0.$$

Applying Chebyshev's inequality,

$$\Pr[T \geq \tau] \leq \frac{\mathrm{Var}(T)}{\tau^2} \leq \frac{\ell}{56\, n(n-1)\tau^2}.$$

Using the threshold

$$\tau = \frac{1}{2} \cdot \frac{1}{6400} \cdot \frac{N}{n(n-1)} \cdot \frac{\ell\varepsilon^2}{7d},$$

we obtain

$$\Pr[T \geq \tau] = O\left(\frac{d^2 n(n-1)}{N^2 \ell \varepsilon^4}\right) < \frac{1}{100},$$

whenever

$$\frac{N}{n} > C_1 \cdot \frac{d}{\sqrt{\ell}\varepsilon^2}$$

for a sufficiently large absolute constant $C_1 > 0$.

Case 2: $\|\mu\|_2 \geq \varepsilon$. Using standard properties of Erfc,

$$|\tilde{p}_i^{(k)}| \geq \frac{1}{8}\sqrt{\frac{m_k}{7}}\,|\tilde{\mu}_i|,$$

Therefore,

$$\begin{aligned}
\mathbb{E}[T] &= \frac{1}{n(n-1)} \sum_{i=1}^{\ell/7} \sum_{k_1 \neq k_2} \tilde{p}_i^{(k_1)} \tilde{p}_i^{(k_2)} \\
&\geq \frac{1}{64\, n(n-1)} \sum_{i=1}^{\ell/7} \sum_{k_1 \neq k_2} \sqrt{\frac{m_{k_1}}{7}}\sqrt{\frac{m_{k_2}}{7}}\,\tilde{\mu}_i^2 \\
&= \frac{N}{64\, n(n-1)} \|\tilde{\boldsymbol{\mu}}\|_2^2.
\end{aligned}$$

By Lemma 11, with probability at least $8/25$,

$$\|\tilde{\boldsymbol{\mu}}\|_2^2 \geq \frac{1}{100} \cdot \frac{\ell}{7d} \cdot \varepsilon^2.$$

Hence, with probability at least $8/25$,

$$\mathbb{E}[T] \geq \frac{1}{6400} \cdot \frac{N}{n(n-1)} \cdot \frac{\ell\varepsilon^2}{7d}.$$

Conditioned on this event, applying Chebyshev's inequality,

$$\begin{aligned}
\Pr[T \leq \tau] &= \Pr\left(\mathbb{E}[T] - T \geq \frac{\mathbb{E}[T]}{2}\right) \\
&\leq \frac{4\,\mathrm{Var}(T)}{\mathbb{E}[T]^2} \\
&= O\left(\frac{d^2 n^2}{N^2 \ell \varepsilon^4} + \frac{nd}{N\ell\varepsilon^2}\right).
\end{aligned}$$

Thus,

$$\Pr[T \le \tau] < \frac{1}{100},$$

provided

$$\frac{N}{n} > C_2 \cdot \frac{d}{\sqrt{\ell}\varepsilon^2}$$

for a sufficiently large absolute constant $C_2 > 0$.

Therefore, a single repetition accepts with probability at least 99/100, if $\|\boldsymbol{\mu}\|_2 = 0$ and rejects with probability at least $1 - \left(\frac{17}{25} + \frac{8}{25} \cdot \frac{1}{100}\right) > \frac{3}{10}$, if $\|\boldsymbol{\mu}\|_2 \ge \varepsilon$. Repeating independently 7 times and accepting only if all repetitions output accept reduces the error probability below 1/10. Finally, by Remark 12, the 7 independent BRHTs require only $O(\log(d/\ell))$ shared random bits. ◀

▶ Remark 17. Note that for a fixed sampling budget $m = \sum_{k\in[n]} m_k$, $\sum_{k_1\neq k_2\in[n]} \sqrt{m_{k_1}}\sqrt{m_{k_2}}$ is maximized when $m_{k_1} = m_{k_2} = m/n$ for all $k_1, k_2 \in [n]$. Then, we need $m = O(d/\sqrt{\ell}\varepsilon^2)$.

6 Heterogeneous communication

Having addressed limited (public) randomness and heterogeneous sample sizes in the previous two sections, we now turn to the last generalization considered in this work, heterogeneous communication budgets. Specifically, we will establish Theorem 3, restated below:

▶ **Theorem 18** (Theorem 3, restated). *For $\boldsymbol{\ell} = (\ell_1, \dots, \ell_n) \in \mathbb{Z}_+^n$ and $s = \Omega(\log(d/\|\boldsymbol{\ell}\|_\infty))$, there exists a protocol for distributed Gaussian mean testing under $\boldsymbol{\ell}$-bit communication constraint with n users, where user k has one sample and can send ℓ_k bits to the referee, provided that*

$$\frac{\|\boldsymbol{\ell}\|_1}{\sqrt{\|\boldsymbol{\ell}\|_\infty}} \ge C \cdot \frac{d}{\varepsilon^2}$$

where $C > 0$ is an absolute constant.

Proof. In view of the previous protocols, the idea is relatively straightforward: the users use $s/7$ bits of shared randomness to sample a (d, d_s)-BRHT R, for

$$d_s := d/2^{\lfloor s/28 \rfloor}.$$

Let $L := d_s \vee \max_{1\le k\le n} \ell_k = \Theta(\max_{1\le k\le n} \ell_k)$ (the last equality by our assumption on s) and $\tilde{\boldsymbol{\mu}} = (R\boldsymbol{\mu})_{[1:L]}$. As in the proof of Theorem 1, each user applies R to its sample to get $\tilde{\boldsymbol{X}}^{(k)} = (R\boldsymbol{X}^{(k)})_{[1:L]}$, so that $\tilde{\boldsymbol{X}}^{(k)} \sim \mathcal{G}(\tilde{\boldsymbol{\mu}}, \mathbb{I}_L)$. Using Lemma 11, with probability at least 8/25,

$$\|\tilde{\boldsymbol{\mu}}\|_2^2 \ge \frac{1}{100} \cdot \frac{L}{d} \cdot \|\boldsymbol{\mu}\|_2^2$$

so that in one case $\tilde{\boldsymbol{\mu}} = \mathbf{0}$ and in the other $\|\tilde{\boldsymbol{\mu}}\|_2^2 = \Omega(\frac{L\varepsilon^2}{d})$ with constant probability. This is where we depart from Theorem 1. The users and referee will now use the same idea of distributed simulation, but in a slightly different fashion: recall that all parameters are known to all parties, and that by choice of L we have $\ell_k \le L$ for all k. Then, user 1 sends the first ℓ_1 bits of their resulting L-dimensional sample; user 2 sends the following ℓ_2 bits,

wrapping around (i.e., the bits $\ell_1+1, \ell_1+2, \ldots, \ell_1+\ell_2 \bmod L$), and so on. Overall, the $\ell_1+\cdots+\ell_n = \|\boldsymbol{\ell}\|_1$ bits sent allow the referee to reconstruct exactly $n' := \lfloor \|\boldsymbol{\ell}\|_1/L \rfloor$ binary samples $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_{n'}$ as in the proof of Theorem 8, which leads to a successful protocol whenever

$$n' \geq C \cdot \frac{\sqrt{d_s}}{\varepsilon^2(d_s/d)}$$

i.e.,

$$\frac{\|\boldsymbol{\ell}\|_1}{\|\boldsymbol{\ell}\|_\infty} \geq C \cdot \frac{d}{\varepsilon^2\sqrt{\|\boldsymbol{\ell}\|_\infty}}$$

from our settings of L and d_s (after, as before, including the probability amplification over 7 independent repetitions). ◀

7 Putting it all together: Theorem 4

With protocols for the three settings, we conclude by providing a protocol for the general setting where shared randomness, sample counts and communication budgets are all free parameters. The protocol is relatively straightforward as it combines the protocols for the three settings.

▶ **Theorem 19** (Theorem 4, restated). *For $s \geq 0$, $\boldsymbol{m} = (m_1, \ldots, m_n) \in \mathbb{Z}_+^n$, and $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_n) \in \mathbb{Z}_+^n$, there exists a protocol for distributed Gaussian mean testing with n users sharing s bits of public randomness, where user k has m_k samples and can send ℓ_k bits to the referee, provided that there exists a partition $\mathcal{P} = \{P_1, \ldots, P_K\}$ of the n users such that*

$$\frac{\sum_{j_1 \neq j_2 \in [K]} \sqrt{\min_{i_1 \in P_{j_1}}(m_{i_1}) \cdot \min_{i_2 \in P_{j_2}}(m_{i_2})}}{K} \geq C \cdot \frac{d}{\varepsilon^2\sqrt{\max\left(\frac{d}{2^{\Theta(s)}}, \|\boldsymbol{\ell}\|_\infty\right)}}$$

and, for every $P \in \mathcal{P}$, $\sum_{i \in P} \ell_i \geq r \cdot \max\left(\frac{d}{2^{\Theta(s)}}, \|\boldsymbol{\ell}\|_\infty\right)$ and $\min_{i \in P}(m_i) \in [r, r\varepsilon^{-2}]$; where $C, r > 0$ are absolute constants.

Proof. Let $\mathcal{P} = \{P_1, \ldots, P_K\}$ be a partition of users (for all j_1, j_2 $P_{j_1} \cap P_{j_2} = \emptyset$, $\bigcup_j P_j = [n]$) such that for all j, $\sum_{i \in P_j} \ell_i \geq 7 \cdot \max\left(\frac{d}{2^{\lfloor s/28 \rfloor}}, \|\boldsymbol{\ell}\|_\infty\right)$. Define

$$L := \max\left(\frac{d}{2^{\lfloor s/28 \rfloor}}, \|\boldsymbol{\ell}\|_\infty\right)$$

Using the distributed simulation idea in Theorem 18, users in each $P_j \in \mathcal{P}$ can simulate *one* L-dimensional binary sample. But, users in P_j hold $(m_i)_{i \in P_j}$ samples. Thus, the setting can be reduced to $K = |\mathcal{P}|$ users holding $(\min_{i \in P_1}(m_i), \ldots, \min_{i \in P_K}(m_i))$ samples under L-bit communication constraints, for which we invoke the protocol of Algorithm 2. ◀

References

1 Jayadev Acharya, Clément L. Canonne, Yanjun Han, Ziteng Sun, and Himanshu Tyagi. Domain compression and its application to randomness-optimal distributed goodness-of-fit. In *COLT*, Proceedings of Machine Learning Research, pages 3–40. PMLR, 2020. URL: http://proceedings.mlr.press/v125/acharya20a.html.

2 Jayadev Acharya, Clément L. Canonne, Ziteng Sun, and Himanshu Tyagi. Unified lower bounds for interactive high-dimensional estimation under information constraints. In *NeurIPS*, 2023.

3 Jayadev Acharya, Clément L. Canonne, and Himanshu Tyagi. Distributed signal detection under communication constraints. In *COLT*, Proceedings of Machine Learning Research, pages 41–63. PMLR, 2020. URL: http://proceedings.mlr.press/v125/acharya20b.html.

4 Jayadev Acharya, Clément L. Canonne, and Himanshu Tyagi. Inference under information constraints II: communication constraints and shared randomness. *arXiv preprint*, 66(12):7856–7877, 2020. arXiv:1905.08302.

5 Nir Ailon and Bernard Chazelle. The fast johnsonlindenstrauss transform and approximate nearest neighbors. *SIAM Journal on Computing*, 39:302–322, May 2009. doi:10.1137/060673096.

6 Clément L. Canonne. Topics and techniques in distribution testing: A biased but representative sample. *Found. Trends Commun. Inf. Theory*, 19(6):1032–1198, 2022. doi:10.1561/0100000114.

7 Clement L. Canonne, Xi Chen, Gautam Kamath, Amit Levi, and Erik Waingarten. Random restrictions of high dimensional distributions and uniformity testing with subcube conditioning. In *Proceedings of the Thirty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '21, pages 321–336, USA, 2021. Society for Industrial and Applied Mathematics. doi:10.1137/1.9781611976465.21.

8 Clément L. Canonne, Abigail Gentle, and Vikrant Singhal. Uniformity testing under user-level local privacy. In *ITCS*, LIPIcs, pages 33:1–33:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. doi:10.4230/LIPIcs.ITCS.2026.33.

9 Clément L. Canonne, Themis Gouleakis, Yuhao Wang, and Joy Qiping Yang. Gaussian mean testing under truncation. In *AISTATS*, Proceedings of Machine Learning Research, pages 4879–4887. PMLR, 2025. URL: https://proceedings.mlr.press/v258/canonne25a.html.

10 Ilias Diakonikolas, Daniel M. Kane, and Ankit Pensia. Gaussian mean testing made simple. In *SOSA*, pages 348–352. SIAM, 2023. doi:10.1137/1.9781611977585.CH31.

11 Botond Szabó, Lasse Vuursteen, and Harry van Zanten. Optimal distributed composite testing in high-dimensional gaussian models with 1-bit communication. *IEEE Trans. Inf. Theory*, 68(6):4070–4084, 2022. doi:10.1109/TIT.2022.3150599.

12 Botond Szabó, Lasse Vuursteen, and Harry van Zanten. Optimal high-dimensional and nonparametric distributed testing under communication constraints. *Ann. Statist.*, 51(3):909–934, 2023. doi:10.1214/23-aos2269.

13 Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.

A Deferred Proofs

▶ **Lemma 20.** *Let $Z_1, \dots, Z_n$ be independent random variables such that $Z_k \in \{-1/2, 1/2\}$ and $\mathbb{E}[Z_k] = p_k$. Assume furthermore $p_{k_1} p_{k_2} \geq 0$ for all $k_1 \neq k_2$ and define*

$$T := \sum_{k_1 \neq k_2} Z_{k_1} Z_{k_2}.$$

Then the following holds:

1. $\mathbb{E}[T] = \sum_{k_1 \neq k_2} p_{k_1} p_{k_2}$
2. $\mathrm{Var}(T) \leq \frac{n(n-1)}{8} + (n-2)\mathbb{E}[T]$

Proof. Define

$$S := \sum_{k \in [n]} Z_k.$$

Since $Z_k^2 = 1/4$, we have

$$T = \sum_{k_1 \neq k_2} Z_{k_1} Z_{k_2} = \left(\sum_{k \in [n]} Z_k \right)^2 - \sum_{k \in [n]} Z_k^2 = S^2 - \frac{n}{4}.$$

1. $\mathbb{E}[\boldsymbol{T}]$**.** Using independence,

$$\mathbb{E}[S] = \sum_{k \in [n]} p_k$$

and

$$\mathrm{Var}(S) = \sum_{k \in [n]} \left(\frac{1}{4} - p_k^2 \right).$$

Hence,

$$\begin{aligned} \mathbb{E}[S^2] &= \mathrm{Var}(S) + \mathbb{E}[S]^2 \\ &= \sum_{k \in [n]} \left(\frac{1}{4} - p_k^2 \right) + \left(\sum_{k \in [n]} p_k \right)^2 \\ &= \frac{n}{4} + \sum_{k_1 \neq k_2} p_{k_1} p_{k_2}. \end{aligned}$$

Therefore,

$$\mathbb{E}[T] = \mathbb{E}[S^2] - \frac{n}{4} = \sum_{k_1 \neq k_2} p_{k_1} p_{k_2}.$$

2. $\mathbf{Var}[\boldsymbol{T}]$**.** Since $T = S^2 - n/4$,

$$\mathrm{Var}(T) = \mathrm{Var}(S^2) = \mathbb{E}[S^4] - \mathbb{E}[S^2]^2.$$

Expanding S^4 and using independence together with

$$\mathbb{E}[Z_k] = p_k, \qquad Z_k^2 = \frac{1}{4}, \qquad \mathbb{E}[Z_k^3] = \frac{p_k}{4}, \qquad Z_k^4 = \frac{1}{16},$$

we obtain

$$\begin{aligned} \mathbb{E}[S^4] = \frac{n}{16} &+ \frac{3n(n-1)}{16} \\ &+ \sum_{k_1 \neq k_2} p_{k_1} p_{k_2} + \frac{3(n-2)}{2} \sum_{k_1 \neq k_2} p_{k_1} p_{k_2} \\ &+ \sum_{k_1 \neq k_2 \neq k_3 \neq k_4} p_{k_1} p_{k_2} p_{k_3} p_{k_4}. \end{aligned}$$

On the other hand,

$$\mathbb{E}[S^2]^2 = \frac{n^2}{16} + \frac{n}{2} \sum_{k_1 \neq k_2} p_{k_1} p_{k_2} + \left(\sum_{k_1 \neq k_2} p_{k_1} p_{k_2} \right)^2.$$

Subtracting,

$$\mathrm{Var}(T) = \mathbb{E}[S^4] - \mathbb{E}[S^2]^2 = \frac{n(n-1)}{8} + (n-2)\sum_{k_1 \neq k_2} p_{k_1} p_{k_2} - 4\sum_{k_1 \neq k_2 \neq k_3} p_{k_1}^2 p_{k_2} p_{k_3} - 2\sum_{k_1 \neq k_2} p_{k_1}^2 p_{k_2}^2.$$

Since $p_{k_1} p_{k_2} \geq 0$ for all $k_1 \neq k_2$ and

$$\mathbb{E}[T] = \sum_{k_1 \neq k_2} p_{k_1} p_{k_2},$$

the result follows. ◀

▶ **Theorem 21.** *Algorithm 1 achieves sample complexity $\Theta(\sqrt{d}/\varepsilon^2)$ for testing whether $\boldsymbol{p} = \boldsymbol{u}_d$ versus $\|\boldsymbol{p} - \boldsymbol{u}_d\|_2 \geq \varepsilon$ for a d-dimensional binary product distribution with mean vector $\boldsymbol{p}$.*

Proof. Let $\tilde{\boldsymbol{p}} := \boldsymbol{p} - \boldsymbol{u}_d$. For each coordinate $i \in [d]$, define

$$T_i := \sum_{k_1 \neq k_2 \in [n]} \left(X_i^{(k_1)} - \frac{1}{2}\right)\left(X_i^{(k_2)} - \frac{1}{2}\right),$$

and

$$T := \frac{1}{n(n-1)} \sum_{i=1}^{d} T_i.$$

Fix $i \in [d]$, and define

$$Z_k := X_i^{(k)} - \frac{1}{2}.$$

Then $Z_1, \ldots, Z_n$ are independent random variables taking values in $\{-1/2, 1/2\}$, with $\mathbb{E}[Z_k] = \tilde{p}_i$. Applying Lemma 20,

$$\mathbb{E}[T_i] = n(n-1)\tilde{p}_i^2$$

and

$$\mathrm{Var}(T_i) \leq \frac{n(n-1)}{8} + n(n-1)(n-2)\tilde{p}_i^2.$$

Summing over coordinates,

$$\mathbb{E}[T] = \|\tilde{\boldsymbol{p}}\|_2^2 = \|\boldsymbol{p} - \boldsymbol{u}_d\|_2^2,$$

and

$$\begin{aligned}\mathrm{Var}(T) &= \frac{1}{n^2(n-1)^2} \sum_{i=1}^{d} \mathrm{Var}(T_i) \\ &\leq \frac{d}{8n(n-1)} + \frac{n-2}{n(n-1)} \|\tilde{\boldsymbol{p}}\|_2^2.\end{aligned}$$

We now analyze the two cases:

Case 1: $p = u_d$. In this case, $\mathbb{E}[T] = 0$ and therefore

$$\mathrm{Var}(T) \leq \frac{d}{8n(n-1)}.$$

By Chebyshev's inequality,

$$\Pr\left(T > \frac{\varepsilon^2}{2}\right) \leq \frac{4\,\mathrm{Var}(T)}{\varepsilon^4} \leq O\left(\frac{d}{n^2\varepsilon^4}\right).$$

Case 2: $\|p - u_d\|_2 \geq \varepsilon$. In this case,

$$\mathbb{E}[T] = \|\boldsymbol{p} - \boldsymbol{u}_d\|_2^2 \geq \varepsilon^2.$$

Again by Chebyshev's inequality,

$$\begin{aligned}\Pr\left(T \leq \frac{\varepsilon^2}{2}\right) &= \Pr\left(\mathbb{E}[T] - T \geq \frac{\mathbb{E}[T]}{2}\right)\\ &\leq \frac{4\,\mathrm{Var}(T)}{\mathbb{E}[T]^2}\\ &\leq O\left(\frac{d}{n^2\varepsilon^4} + \frac{1}{n\varepsilon^2}\right).\end{aligned}$$

Thus, choosing

$$n > C \cdot (\sqrt{d}/\varepsilon^2)$$

for some absolute constant $C > 0$, makes both error probabilities smaller than $1/100$. ◀

Hard-To-Sample Distributions from Robust Extractors

Farzan Byramji
UC San Diego, La Jolla, CA, USA

Daniel M. Kane
UC San Diego, La Jolla, CA, USA

Jackson Morris
UC San Diego, La Jolla, CA, USA

Anthony Ostuni
UC San Diego, La Jolla, CA, USA

Abstract

We provide a unified method for constructing explicit distributions which are difficult for restricted models of computation to generate. Our constructions are based on a new notion of *robust extractors*, which are extractors that remain sound even when a small number of points violate the min-entropy constraint. Using such objects, we show that for a broad range of sampling models (e.g., low-depth circuits, small-space sources, etc.), every output of the model has distance $1 - o(1)$ from our target distribution, qualitatively recovering essentially all previously known hardness results. Our work extends that of Viola (SICOMP '14), who developed an earlier unified framework based on traditional extractors to rule out sampling with very small error.

As a further application of our technique, we leverage a recent extractor construction of Chattopadhyay, Goodman, and Gurumukhani (ITCS '24) to present the first explicit distribution with distance $1 - o(1)$ from the output of any low-degree $\mathbb{F}_2$-polynomial source. We note that a similar bound was obtained concurrently and independently by Khodabandeh and Shinkar (ECCC '26). We also describe a potential avenue toward proving a similar hardness result for $\mathsf{AC}^0[\oplus]$ circuits.

2012 ACM Subject Classification Theory of computation → Circuit complexity; Theory of computation → Pseudorandomness and derandomization

Keywords and phrases sampling, extractor, low-degree polynomials

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.37

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2604.26179

Funding *Farzan Byramji*: Supported by Simons Investigator Award #929894, and NSF Awards CCF-2425349 and AF: Medium 2212136.
Daniel M. Kane: Supported by NSF Medium Award CCF-2107547.
Anthony Ostuni: Partially supported by Simons Investigator Award #929894 and NSF Award CCF-2425349.

Acknowledgements We thank Jesse Goodman and Mohit Gurumukhani for a number of helpful comments on an earlier draft. In particular, we are grateful for the suggestion of and discussion about "smooth extractors" as mentioned in Subsection 3.2. We also thank anonymous reviewers for their useful feedback.

1 Introduction

The quest to prove unconditional hardness results for restricted models of computation (e.g., low-depth circuits) is one of the major programs of complexity theory. This program saw a number of major successes in the 1980s when researchers exposed the limitations of AC^0 and $\mathsf{AC}^0[p]$ circuits with simple and explicit hard-to-compute functions [19, 1, 50, 24, 25, 41, 37].


© Farzan Byramji, Daniel M. Kane, Jackson Morris, and Anthony Ostuni;
licensed under Creative Commons License CC-BY 4.0

Unfortunately, the current state of affairs on this front remains rather humble: even four decades later we still cannot rule out the preposterous claim that all of PSPACE can be computed by polynomially-sized AC^0 circuits with mod 6 gates.

It would be disingenuous, however, to suggest that no modern progress has occurred. Even just the past few years have seen impressive developments in understanding the limitations of these models through the lens of pseudorandom generators [16, 17, 27, 32, 33] (see also the survey [26]), extractors [13, 20, 3], and quantum analogs [36, 5, 28, 18, 23].[1]

One particularly exciting line of inquiry is understanding the capabilities of these weak models to generate specific distributions (when taking random bits as input). The landscape here is dramatically different from the traditional task of computing specific functions. For example, if a circuit can compute a function $f\colon \{0,1\}^n \to \{0,1\}$, then it can output the uniform distribution over $(x, f(x))$ by simply computing f on each random input. The converse, however, is not true. While PARITY has long been known to be difficult for AC^0 circuits to compute [19, 1, 50, 24, 25, 41, 37], one can produce the uniform distribution over input-output pairs by simply mapping the uniformly random bits $(x_1, x_2, \ldots, x_{n+1})$ to $(x_1 \oplus x_2, x_2 \oplus x_3, \ldots, x_n \oplus x_{n+1}, x_{n+1} \oplus x_1)$ [6, 11].

Even still, for many common models including small AC^0 circuits [35, 8], small-space sources [14], and communication protocols [4, 21, 14, 52], researchers have discovered explicit distributions which have total variation distance $1 - o(1)$ from any distribution produced by the model. In comparison to our knowledge of computation, the glaring shortcoming in our understanding of sampling is that we do not know of any explicit distribution which cannot be (approximately) produced by polynomially-sized AC^0 circuits with mod p gates for any prime p.

Beyond its inherent intrigue, the complexity of distributions has intimate connections with data structure lower bounds [44, 35, 8, 48, 14, 49, 52, 29, 2], quantum supremacy [10, 29, 22], explicit codes [40], and learning theory [30]. Moreover, the field has had a fruitful relationship with the study of *extractors* [42, 45, 15, 46, 7, 39], which are objects that convert particular sources of randomness into approximately uniform ones.

This work further develops the latter connection by introducing a general paradigm based on *robust extractors* to prove sampling lower bounds. Intuitively, one can view robust extractors as extractors which remain sound even when a small number of points violate the min-entropy condition. There are several ways to formalize this notion, and we defer our exact definition and discussion of alternatives to Subsection 3.2. For now, we will say a robust extractor $\mathsf{rExt}\colon \{0,1\}^n \to \{0,1\}^m$ for a class of distributions $\mathcal{X}$ satisfies the following two conditions:

1. (Extractor Property) For every source $\mathbf{X} \in \mathcal{X}$ with sufficient min-entropy, the output of rExt on $\mathbf{X}$ is close in total variation (TV) distance to the uniform distribution over $\{0,1\}^m$, and
2. (Robustness Property) For every source $\mathbf{X} \in \mathcal{X}$, the probability that $\mathbf{x} \sim \mathbf{X}$ lands in the set of $\mathbf{X}$'s low probability points and $\mathsf{rExt}(\mathbf{x}) = 0^m$ is not much more than 2^{-m}.

With these objects in hand, we can generically prove strong sampling lower bounds. This result further advances a similar approach based on traditional extractors [46], which we discuss in greater detail in Section 2. Below, $\mathbf{U}^n$ denotes the uniform distribution over $\{0,1\}^n$, and for a function $f\colon \{0,1\}^n \to \{0,1\}^m$, $f(\mathbf{U}^n)$ denotes the output distribution of f with n uniformly random bits as input.

[1] This is only a minute fraction of contemporary work, and we encourage the interested reader to consult references within those cited.

▶ **Theorem 1** (Informal version of Theorem 8). *Let $\mathcal{X}$ be a class of distributions over $\{0,1\}^{t(n+1)}$, and let $\mathcal{Y}$ be the class of distributions over $\{0,1\}^n$ obtained from "low complexity" functions of $\mathcal{X}$. Suppose $\mathsf{rExt}\colon \{0,1\}^n \to \{0,1\}^m$ is a sufficiently good robust extractor for $\mathcal{Y}$, and define the function $f\colon \{0,1\}^n \to \{0,1\}$ by $f(x) = \mathbf{1}(\mathsf{rExt}(x) = 0^m)$. If we consider the distribution*

$$\mathbf{D} = (\mathbf{U}_1, \mathbf{U}_2, \dots, \mathbf{U}_t, f(\mathbf{U}_1), f(\mathbf{U}_2), \dots, f(\mathbf{U}_t)),$$

where $\mathbf{U}_1, \dots, \mathbf{U}_t$ are independent copies of $\mathbf{U}^n$, then every source $\mathbf{X} \in \mathcal{X}$ has TV distance

$$\|\mathbf{X} - \mathbf{D}\|_{\mathsf{TV}} \geq 1 - o_{t,n}(1),$$

where $o_{t,n}(1) \to 0$ as $t, n \to \infty$.

Moreover, if $\widetilde{\mathsf{rExt}}\colon \{0,1\}^n \to \{0,1\}$ is a sufficiently good robust extractor (in a different parameter regime) for $\mathcal{Y}$, then $\left\|\mathbf{X} - (\mathbf{U}^n, \widetilde{\mathsf{rExt}}(\mathbf{U}^n))\right\|_{\mathsf{TV}} \geq \frac{1}{4} - o_n(1)$.

▶ Remark 2. Robust extractors are the motivating object behind our approach, but actually a weaker one-sided notion in place of the typical extractor guarantee (1) suffices for our purposes. In particular, Theorem 1 holds as long as rExt on the uniform distribution assigns decent probability to 0^m. The details can be found in Section 2.

We highlight that the "moreover" part of Theorem 1 can be viewed as a stronger type of lower bound than those for approximate computation. If, for example, the functions $f, g\colon \{0,1\}^n \to \{0,1\}$ agree on a $(1-\delta)$-fraction of their inputs, then it is easy to generate the uniform distribution over $(\mathbf{U}^n, g(\mathbf{U}^n))$ to TV distance at most δ by simply computing f on each random input.

Using Theorem 1, we can recover unconditional hardness results for sampling in essentially every model where they are currently known (albeit with weaker decay rates). We are also able to provide the first explicit[2] distribution which has distance $1 - o(1)$ from the output of any low-degree $\mathbb{F}_2$-polynomial source, making progress toward a similar result for $\mathsf{AC}^0[\oplus]$ circuits.[3] Previous arguments [46, 13] only forbade distance $2^{-\Omega(n)}$.

▶ **Theorem 3** (Informal instantiations of Theorem 1). *Let $\mathcal{X}$ be one of the following classes of sources over $\{0,1\}^N$:*

- *low-degree $\mathbb{F}_2$-polynomial,*
- *local (NC^0),*
- *circuit (AC^0) ,*
- *communication,*
- *small-space,*
- *Turing machine.*

Then every source $\mathbf{X} \in \mathcal{X}$ satisfies

$$\|\mathbf{X} - \mathbf{D}\|_{\mathsf{TV}} \geq 1 - o(1),$$

where $\mathbf{D}$ is defined by taking many independent copies of $(\mathbf{U}^n, \mathbf{1}(\mathsf{rExt}(\mathbf{U}^n) = 0^m))$ for an explicit robust extractor $\mathsf{rExt}\colon \{0,1\}^n \to \{0,1\}^m$ (depending on the choice of $\mathcal{X}$).

[2] One can essentially use a standard counting argument to show such a distribution exists; see the appendix in the full version.

[3] Recall these are AC^0 circuits with mod 2 gates.

Moreover, there exists an explicit robust extractor $\widetilde{\mathsf{rExt}}\colon \{0,1\}^{N-1} \to \{0,1\}$ *(in a different parameter regime) such that every source* $\mathbf{X} \in \mathcal{X}$ *satisfies*

$$\left\|\mathbf{X} - (\mathbf{U}^{N-1}, \widetilde{\mathsf{rExt}}(\mathbf{U}^{N-1}))\right\|_{\mathsf{TV}} \geq \frac{1}{4} - o(1).$$

We emphasize that bounds of this form were already known (with better quantitative behavior) for local and circuit [35, 8], communication [4, 21, 14, 52], small-space [14], and Turing-machine sources [45, 14]. Our novel contribution is obtaining these bounds in a *unified* way, as well as providing strong bounds for low-degree $\mathbb{F}_2$-polynomial sources.

Concurrent Work

In independent and concurrent work, Khodabandeh and Shinkar also investigate the complexity of sampling with low-degree $\mathbb{F}_2$-polynomials [31]. Using very different techniques from our own, they prove that the output of any such polynomial sampler has TV distance $1 - o(1)$ from the $(1/3)$-biased product distribution. As in the present work, the $o(1)$ term has suboptimal decay, but in the case of degrees 1, 2, and 3, they prove it can be taken to be $\exp(-\Omega(N))$, $\exp(-\Omega(\log(N)/\log\log(N)))$, and $\exp(-\Omega(\sqrt{\log\log(N)}))$, respectively, which is stronger than we are able to obtain via our robust extractor framework. (Note, however, that an extractor-based approach can match their bound for degree-1 polynomials – see Section 5.)

Paper Organization

We state and prove Theorem 8, the precise version of Theorem 1, in Section 2, along with additional background and context. We then review some preliminary material in Section 3 before instantiating Theorem 8 in Section 4 to obtain Theorem 3 for polynomial sources. We address the other classes (local, circuit, communication, small-space, and Turing-machine sources) in the full version. We conclude with some open problems in Section 5. Supplementary material on non-explicit constructions can be found in the appendix of the full version.

2 Background and Main Result

In this section, we state and prove our main result, Theorem 8. Before doing so, it will be instructive to review existing arguments and their limitations to develop intuition for our approach. For concreteness, we will focus our discussion on distributions generated by low-degree $\mathbb{F}_2$-polynomials, but essentially all of the analysis holds more generally. The meaning of any unfamiliar terminology or notation in this section can be found in Section 3.

Let $P\colon \mathbb{F}_2^r \to \mathbb{F}_2^n$ be a polynomial map defined by n arbitrary degree-d polynomials $\{p_i\colon \mathbb{F}_2^r \to \mathbb{F}_2\}_{i=1}^n$ acting on the same r input bits, where we view r as some arbitrarily large integer. Our goal is to construct an explicit distribution $\mathbf{D}$ over $\{0,1\}^n$ such that regardless of how the p_i's are defined, the output distribution of P on r uniformly random bits, denoted $P(\mathbf{U}^r)$, has total variation distance at least $1 - o(1)$ from $\mathbf{D}$. (Here, we identify $\mathbb{F}_2^n$ with $\{0,1\}^n$.)

2.1 A Nonzero Lower Bound

We begin with the more modest goal of showing that P cannot *exactly* generate some explicit distribution $\mathbf{D}$. Here, we may invoke an argument of Viola [46, 47] based on extractors, whose formal definition we now recall.

▶ **Definition 4** (Extractor). *A function* $\mathsf{Ext}\colon \{0,1\}^n \to \{0,1\}$ *is an* (ε, k)-extractor *for a class* $\mathcal{X}$ *of distributions over* $\{0,1\}^n$ *if for every source* $\mathbf{X} \in \mathcal{X}$ *with min-entropy at least* k*, we have*

$$\mathbf{Pr}[\mathsf{Ext}(\mathbf{X}) = 1] = \frac{1}{2} \pm \varepsilon.$$

We will take the hard distribution to be uniform over input-output pairs of an explicit (ε, k)-extractor $\mathsf{Ext}\colon \{0,1\}^{n-1} \to \{0,1\}$ for polynomial sources of degree $2d$, where ε is some small constant and $k = n - O(1)$. That is, $\mathbf{D} = (\mathbf{U}^{n-1}, \mathsf{Ext}(\mathbf{U}^{n-1}))$. Such extractors which are computable in time $\mathsf{poly}(n)$ are known [13]. For clarity, we express $P(\mathbf{U}^r)$ similarly as $(Q(\mathbf{U}^r), q(\mathbf{U}^r))$, where $Q\colon \{0,1\}^r \to \{0,1\}^{n-1}$ and $q\colon \{0,1\}^r \to \{0,1\}$ are degree-d polynomials acting on the same set of input bits, corresponding to the first $n-1$ output bits and last output bit of P, respectively.

Suppose for contradiction that $P(\mathbf{U}^r)$ exactly generates $\mathbf{D}$. Viola's argument considers the random variable $\mathbf{M}$ over $\{0,1\}^{n-1}$ defined by sampling $\mathbf{u} \sim \mathbf{U}^r$ and outputting $Q(\mathbf{u})$ if $q(\mathbf{u}) = 1$, and otherwise $n-1$ uniformly random bits independent of $\mathbf{U}^r$, which we denote by $\widetilde{\mathbf{U}}^{n-1}$. Written suggestively similar to a polynomial, we have

$$\mathbf{M} = q(\mathbf{U}^r)Q(\mathbf{U}^r) + (1 - q(\mathbf{U}^r))\widetilde{\mathbf{U}}^{n-1}, \tag{1}$$

or equivalently by our assumption,

$$\mathbf{M} = \mathsf{Ext}(\mathbf{U}^{n-1})\mathbf{U}^{n-1} + (1 - \mathsf{Ext}(\mathbf{U}^{n-1}))\widetilde{\mathbf{U}}^{n-1}. \tag{2}$$

Now consider the effect of applying Ext to $\mathbf{M}$. By (1), $\mathbf{M}$ can be generated by a degree-$(2d)$ $\mathbb{F}_2$-polynomial source with $n - O(1)$ bits of min-entropy (recall $Q(\mathbf{U}^r)$ is assumed to be uniform), so $\mathbf{Pr}[\mathsf{Ext}(\mathbf{M}) = 1] \le \frac{1}{2} + \varepsilon$. By contrast, whenever the sample $\mathbf{u} \sim \mathbf{U}^{n-1}$ satisfies $\mathsf{Ext}(\mathbf{u}) = 1$, we must have $\mathsf{Ext}(\mathbf{M}) = 1$ by (2), so

$$\begin{aligned}
\mathbf{Pr}[\mathsf{Ext}(\mathbf{M}) = 1] &= \mathbf{Pr}\left[\mathsf{Ext}(\mathbf{M}) = 1 \,\middle|\, \mathsf{Ext}(\mathbf{U}^{n-1}) = 1\right] \cdot \mathbf{Pr}[\mathsf{Ext}(\mathbf{U}^{n-1}) = 1] \\
&\quad + \mathbf{Pr}\left[\mathsf{Ext}(\mathbf{M}) = 1 \,\middle|\, \mathsf{Ext}(\mathbf{U}^{n-1}) = 0\right] \cdot \mathbf{Pr}[\mathsf{Ext}(\mathbf{U}^{n-1}) = 0] \\
&= 1 \cdot \mathbf{Pr}[\mathsf{Ext}(\mathbf{U}^{n-1}) = 1] + \mathbf{Pr}[\mathsf{Ext}(\widetilde{\mathbf{U}}^{n-1}) = 1] \cdot \mathbf{Pr}[\mathsf{Ext}(\mathbf{U}^{n-1}) = 0] \\
&\approx \frac{1}{2} + \frac{1}{4} \gg \frac{1}{2} + \varepsilon,
\end{aligned}$$

a contradiction. That is, no low-degree polynomial can exactly sample $\mathbf{D}$. In fact, a more careful analysis forbids P from sampling $\mathbf{D}$ to distance better than $2^{-\Omega(k)}$ (which is $2^{-\Omega(n)}$ for known explicit constructions of such extractors [13]).

2.2 A Constant Lower Bound

It is initially unclear how to strengthen the previous argument to obtain even some constant distance lower bound. The primary issue is that even if $\mathbf{D}$ has large min-entropy, it might be reasonably close to some distribution $P(\mathbf{U}^n)$ with small min-entropy, so we do not have any guarantees on the output of the extractor on $\mathbf{M}$.

2.2.1 Min-Entropy Polarization

One approach that was successful in the analysis of distributions generated by low-depth circuits is *min-entropy polarization* [48]. Using random restrictions and hypercontractivity, Viola proved that any distribution $C(\mathbf{U}^r)$ produced by a small circuit $C\colon \{0,1\}^r \to \{0,1\}^n$

could be approximated by a not-too-large collection of restrictions of C whose output was either constant or had large min-entropy, in which case an extractor Ext could be meaningfully applied (see Section 4.2 in the full version for additional details). The event witnessing the total variation distance can then be defined as the union of a small set of outputs, corresponding to restrictions under which the circuit is constant, and pairs (x, b) where $\mathsf{Ext}(x) \neq b$.

Unfortunately, one cannot always polarize the output distribution of an arbitrary sampler in this fashion; consider the following example from [48].

▶ **Example 5.** Let $\mathbf{G}$ be the distribution over $\{0,1\}^n$ obtained by sampling a random string $\mathbf{u} \sim \mathbf{U}^n$, and outputting $\mathbf{u}$ if $|\mathbf{u}| \bmod 2 = 1$ and otherwise outputting 0^n. This sampler cannot have its min-entropy polarized via restrictions for the following reason. For any restriction which leaves some variable free, the min-entropy remains 1. This means that every restriction in a polarizing collection must fix every variable, but then $\Omega(2^n)$ many such restrictions are needed to approximate the distribution.

Note, however, that this sampling procedure *can* be implemented by the degree-2 $\mathbb{F}_2$-polynomial map

$$P(y) = \left(\sum_{i=1}^{n} y_i\right) \cdot (y_1, \ldots, y_n) + \left(1 - \sum_{i=1}^{n} y_i\right) \cdot 0^n$$

to express $\mathbf{G}$ as $P(\mathbf{U}^n)$. Hence, we are not able to apply this framework for our purposes.

2.2.2 Discarding Heavy Points

An alternative approach, still somewhat in the spirit of [48], was taken by Chattopadhyay, Goodman, and Zuckerman [14] when considering communication sources (see Section 4.3.1 in the full version for a formal definition). They also defined their witness event to be the union of a small "bad" set and input-output pairs which contradict a particular extractor's definition. Unlike Viola, however, they define their bad set to be strings which are assigned substantial probability by the sampler (excluding the last bit). Since there cannot be many "heavy" strings, this set is indeed small.

The downside to this approach is that the extractor one requires does not necessarily correspond to an extractor for the class of sources being considered. In particular, the argument requires an extractor that works on the source after conditioning on membership in some set. For communication sources, one can reduce to the case of being able to apply a two-source extractor, but it is unclear how to generically perform a similar reduction to known extractors in the case of other sources, such as those generated by low-degree polynomials.

2.2.3 Robust Extractors

To overcome the previous two obstacles, we ask for more from the *extractor* rather than from the *source*. Recall that the issue with trying to strengthen the argument from Subsection 2.1 is that the generated distribution $P(\mathbf{U}^r) = (Q(\mathbf{U}^r), q(\mathbf{U}^r))$ might assign too much mass to a small set of "bad" points, so the auxiliary distribution of $\mathbf{M}$ does not have enough min-entropy to apply an extractor. Our solution is to simply ask for an extractor whose soundness holds on the light (i.e., low probability) points, rather than the entire domain. In other words, we seek an extractor which is robust to a small number of points violating the min-entropy constraint.

To be a bit more precise, suppose that the (ε, k)-extractor $\mathsf{Ext}\colon \{0,1\}^{n-1} \to \{0,1\}$ we have been considering had the property that every degree-$(2d)$ $\mathbb{F}_2$-polynomial source $\mathbf{X}$ satisfies

$$\mathbf{Pr}[\mathbf{X} \in L \text{ and } \mathsf{Ext}(\mathbf{X}) = 1] \leq \frac{1}{2} + \varepsilon, \tag{3}$$

where $L = \{x \in \{0,1\}^{n-1} : \mathbf{Pr}[\mathbf{X} = x] \leq 2^{-k}\}$ is the set of light points.[4] (Recall that a "traditional" extractor is only guaranteed to satisfy (3) when $L = \{0,1\}^{n-1}$.) A priori, it is not obvious that these objects even exist, but it turns out that a straightforward modification to the extractor construction of [13] yields such an object.

In general, we believe that essentially any natural class of distributions should have extractors with a similar guarantee, and much of the present work is devoted to showing that existing extractor constructions (or simple modifications of them) are robust. In fact, one can interpret the aforementioned min-entropy polarization results in [48] as saying that any extractor for circuit sources enjoys the robustness property (see Section 4.2.1 in the full version).

Now that we have a *robust extractor*, we can run a version of our earlier argument. Once again, let $\mathbf{M}$ be the random variable over $\{0,1\}^{n-1}$ defined by sampling $\mathbf{u} \sim \mathbf{U}^r$ and outputting $Q(\mathbf{u})$ if $q(\mathbf{u}) = 1$, and otherwise $n-1$ random bits using fresh randomness. We also define $\widetilde{\mathbf{M}}$ similarly, but with $\mathbf{D} = (\mathbf{U}^{n-1}, \mathsf{Ext}(\mathbf{U}^{n-1}))$ in place of $P(\mathbf{U}^r)$. By (3), we morally have that $\mathsf{Ext}(\mathbf{M})$ is near balanced on its light points, whereas by construction, $\mathsf{Ext}(\widetilde{\mathbf{M}})$ is likely to output 1 on such points. Working out the details (see Subsection 2.4), one finds that $P(\mathbf{U}^r)$ must be $\Omega(1)$-far from $\mathbf{D}$, as desired.

2.3 A $1 - o(1)$ Lower Bound

It remains to amplify the total variation distance lower bound from constant to $1 - o(1)$. Toward this end, we will take many independent copies of the distribution above, and prove that generating this product of distributions is much harder than generating any individual distribution. We note that such *direct product theorems* have been examined previously in the sampling context [14, 22].

Proceeding more formally, we redefine the distribution $\mathbf{D}$ to be

$$\mathbf{D} = (\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_t, \mathsf{Ext}(\mathbf{U}_1), \mathsf{Ext}(\mathbf{U}_2), \ldots, \mathsf{Ext}(\mathbf{U}_t)),$$

where the $\mathbf{U}_i$'s are independent copies of $\mathbf{U}^{n-1}$ and $t \geq 1$ is an integer to be chosen later. Observe that now $\mathbf{D}$ is over $\{0,1\}^N$ for $N \coloneqq tn$, so we must redefine our polynomial map P to also be on N output bits. We emulate much of our previous analysis with the auxiliary random variable $\widetilde{\mathbf{M}}$ generalized to output $\mathbf{u}_i$ if $\mathsf{Ext}(\mathbf{u}_i) = 1$ for any $i \in \{1, \ldots, t\}$ (and $\mathbf{M}$ similarly generalized with respect to $P(\mathbf{U}^r)$). This greatly increases the probability that $\mathsf{Ext}(\widetilde{\mathbf{M}}) = 1$, which further improves our distance bound.

Unfortunately, the approach as written does not provide distance approaching 1. The problem can essentially be traced back to a loss of $\mathbf{Pr}[\mathbf{X} \in L \text{ and } \mathsf{Ext}(\mathbf{X}) = 1]$ coming from our test event. Since the set of low probability points L might be the entire domain, we cannot upper bound this quantity by anything appreciably better than $1/2$ without violating the extractor guarantee. In order to make further progress, we extend our extractor to have multiple output bits.

[4] There are other ways to capture a similar notion of robustness, but this definition seems to be particularly advantageous; see the discussion in Subsection 3.2.

This multi-output extractor $\mathsf{Ext}\colon \{0,1\}^{n-1} \to \{0,1\}^m$ is defined similarly to the single-output version (Definition 4), only now with the pseudorandomness condition being a bound on the total variation distance $\|\mathsf{Ext}(\mathbf{X}) - \mathbf{U}^m\|_{\mathsf{TV}} \leq \varepsilon$. One can again obtain a robust version (with $m = \Omega(\log \log n)$) by modifying the construction of [13]; we defer the formal definition to Subsection 3.2.

The upshot is that now we can replace the bottleneck of the previous argument with an analysis of $\mathbf{Pr}[\mathbf{X} \in L \text{ and } \mathsf{Ext}(\mathbf{X}) = 0^m]$. This, in turn, can be more tightly bounded by $2^{-m} + \varepsilon$. In our construction, ε will tend to 0 as m grows,[5] so working out the calculations, we obtain a bound roughly of the form

$$1 - \exp(-2^{-m} \cdot t) - 2^{-\Omega(n)} \cdot \mathsf{poly}(t)$$

for sufficiently large n. Setting t to be slightly larger than 2^m, we can finally obtain our desired $1 - o(1)$ bound. The full details can be found in the subsequent subsection.

2.3.1 Isolators

Before formalizing our discussion into a theorem, it is worth reflecting on exactly what properties of the robust extractor we used. (These properties may be further illuminated by consulting the proof of Theorem 8.) In analyzing our target distribution $\mathbf{D}$, all that we truly needed is for some output $y \in \{0,1\}^m$ to be assigned substantial mass by $\mathsf{Ext}(\mathbf{U}^{n-1})$. By contrast, our analysis of the generated distribution $P(\mathbf{U}^r)$ only required that the extractor did not map too many of the light points to the same output y. In other words, our argument does not require the full power of robust extractors. We distill the properties we need into the following weaker object, which we call an *isolator*, since it in some sense controls the isolated light points.

▶ **Definition 6** (Isolator). *A function* $\mathsf{Iso}\colon \{0,1\}^n \to \{0,1\}$ *is an* (α, β, k)-isolator *for a class* $\mathcal{X}$ *of distributions over* $\{0,1\}^n$ *if*

1. $\mathbf{Pr}[\mathsf{Iso}(\mathbf{U}^n) = 1] \geq \alpha$, *and*
2. *Every source* $\mathbf{X} \in \mathcal{X}$ *satisfies* $\mathbf{Pr}_{x \sim \mathbf{X}}[x \in L \text{ and } \mathsf{Iso}(x) = 1] \leq \beta$, *where* $L = \{x \in \{0,1\}^n : \mathbf{Pr}[\mathbf{X} = x] \leq 2^{-k}\}$.

It is perhaps worth highlighting that although all our constructions of isolators essentially work by "robustifying" existing extractor constructions, this is not strictly necessary to satisfy Definition 6. It is plausible one could explicitly construct an isolator for $\mathsf{AC}^0[\oplus]$ sources, for example, without first explicitly constructing seemingly elusive extractors for that class.

2.4 Main Result

We conclude Section 2 by stating our main result, Theorem 8, and its proof. Below, the following notation will be convenient for generalizing the random variable $\mathbf{M}$ considered above.

[5] We had previously defined ε to be a small constant, but our construction of robust extractors allows for this smaller setting.

▶ **Definition 7** (Randomized Addressing). *For positive integers t and n, we define the randomized function* $\mathsf{addr}\colon (\{0,1\}^n)^t \times \{0,1\}^t \to \{0,1\}^n$ *as*

$$\mathsf{addr}(A_1, A_2, \ldots, A_t, b_1, b_2, \ldots, b_t) = \begin{cases} A_1 & \text{if } b_1 = 1, \\ A_2 & \text{if } b_1 = 0, b_2 = 1, \\ A_3 & \text{if } b_1 = b_2 = 0, b_3 = 1, \\ \quad\vdots & \\ A_t & \text{if } b_1 = b_2 = \cdots = b_{t-1} = 0, b_t = 1, \\ \mathbf{U}^n & \text{if } b_1 = b_2 = \cdots = b_t = 0. \end{cases}$$

Though the parameters n and t will mostly be clear from context, we will sometimes write $\mathsf{addr}_{n,t}$ for clarity.

We can now finally present our main result.

▶ **Theorem 8.** *Let $\alpha, \beta \in [0,1]$ and t, k, n be positive integers where $k \le n-1$. Let $\mathcal{X}$ be a class of distributions over $\{0,1\}^{t(n+1)}$, and let $\mathcal{Y}$ be the class of distributions over $\{0,1\}^n$ of the form $\mathsf{addr}_{n,t}(\mathbf{X})$ for some $\mathbf{X} \in \mathcal{X}$. Suppose $\mathsf{Iso}\colon \{0,1\}^n \to \{0,1\}$ is an (α, β, k)-isolator for $\mathcal{Y}$, and define the distribution*

$$\mathbf{D} = (\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_t, \mathsf{Iso}(\mathbf{U}_1), \mathsf{Iso}(\mathbf{U}_2), \ldots, \mathsf{Iso}(\mathbf{U}_t)),$$

where $\mathbf{U}_1, \ldots, \mathbf{U}_t$ are independent copies of $\mathbf{U}^n$. Then every source $\mathbf{X} \in \mathcal{X}$ satisfies

$$\|\mathbf{X} - \mathbf{D}\|_{\mathsf{TV}} \ge 1 - (1-\alpha)^{t+1} - 2^{-(n-k)}\left(2t^3 + 1\right) - \beta.$$

In particular, $\|\mathbf{X} - (\mathbf{U}^n, \mathsf{Iso}(\mathbf{U}^n))\|_{\mathsf{TV}} \ge 2\alpha - \alpha^2 - 2^{-(n-k-2)} - \beta$.

▶ Remark 9. Depending on the sampling model and particular goal, the precise ordering of the coordinates in a target distribution may affect whether or not that distribution can be sampled (e.g., [45]). In the case of Theorem 8, the proof goes through for any permutation of the coordinates of addr and $\mathbf{D}$, although the order may affect whether $\mathsf{addr}(\mathbf{X})$ lies in a particular class.

Proof of Theorem 8. Let $\mathbf{X}$ be any source in $\mathcal{X}$. Viewing $\mathbf{X}$ as $(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_t, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t)$ with the $\mathbf{X}_i$'s and $\mathbf{x}_j$'s over $\{0,1\}^n$ and $\{0,1\}$, respectively, we may apply $\mathsf{addr}_{n,t}$ to $\mathbf{X}$ to obtain the source $\mathbf{Y} = \mathsf{addr}(\mathbf{X}) \in \mathcal{Y}$. Next, define the sets of light points in $\mathbf{X}_i$ for $i = 1, 2, \ldots, t$ and $\mathbf{Y}$ by

$$L_{\mathbf{X}_i} = \{x \in \{0,1\}^n : \mathbf{Pr}[\mathbf{X}_i = x] \le 2^{-(\log(t)+k+1)}\} \text{ and } L_{\mathbf{Y}} = \{y \in \{0,1\}^n : \mathbf{Pr}[\mathbf{Y} = y] \le 2^{-k}\}.$$

Observe that $\bigcap_i L_{\mathbf{X}_i} \subseteq L_{\mathbf{Y}}$, since any string $z \in \bigcap_i L_{\mathbf{X}_i}$ satisfies

$$\begin{aligned} \mathbf{Pr}[\mathbf{Y} = z] = \Pr_{\mathbf{X}}[\mathsf{addr}(\mathbf{X}) = z] &\le \mathbf{Pr}[\mathbf{U}^n = z] + \sum_{i=1}^{t} \mathbf{Pr}[\mathbf{X}_i = z] \qquad \text{(by union bound)} \\ &\le 2^{-n} + t \cdot 2^{-(\log(t)+k+1)} \le 2^{-k}. \end{aligned}$$

Thus, the isolator property of Iso guarantees that

$$\begin{aligned} \mathcal{E}(\mathbf{X}) &\coloneqq \Pr_{\mathbf{X}}\left[\bigcap_i \left(\mathbf{X}_i \in \bigcap_j L_{\mathbf{X}_j}\right) \text{ and } \mathsf{Iso}(\mathsf{addr}(\mathbf{X})) = 1\right] \\ &\le \Pr_{\mathbf{X}}[\mathsf{addr}(\mathbf{X}) \in L_{\mathbf{Y}} \text{ and } \mathsf{Iso}(\mathsf{addr}(\mathbf{X})) = 1] + \Pr_{\mathbf{X}}\left[\mathsf{addr}(\mathbf{X}) \notin L_{\mathbf{Y}} \,\middle|\, \bigcap_i \left(\mathbf{X}_i \in \bigcap_j L_{\mathbf{X}_j}\right)\right] \\ &\le \beta + \mathbf{Pr}[\mathbf{U}^n \notin L_{\mathbf{Y}}] \le \beta + 2^{-(n-k)}, \end{aligned} \tag{4}$$

where the final inequality follows from the fact that $\mathbf{Y}$ can assign 2^{-k} probability mass to at most 2^k points.

We will choose $\mathcal{E}$ to be the test witnessing the claimed TVD between $\mathbf{X}$ and $\mathbf{D}$, although it will be slightly more convenient in this part of the analysis to consider the complement event. In a similar fashion to $\mathbf{X}$, we view $\mathbf{D}$ as $(\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_t, \mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_t)$, where recall each $\mathbf{D}_i$ is an independent copy of the uniform distribution $\mathbf{U}^n$. By the union bound, we have

$$\begin{aligned}
&\Pr_{\mathbf{D}}\left[\bigcup_i \left(\mathbf{D}_i \notin \bigcap_j L_{\mathbf{X}_j}\right) \text{ or } \mathsf{Iso}(\mathsf{addr}(\mathbf{D})) = 0\right] \\
&\leq \sum_i \Pr_{\mathbf{D}}\left[\mathbf{D}_i \notin \bigcap_j L_{\mathbf{X}_j}\right] + \Pr_{\mathbf{D}}[\mathsf{Iso}(\mathsf{addr}(\mathbf{D})) = 0] \\
&\leq \sum_{i,j} \Pr_{\mathbf{D}}\left[\mathbf{D}_i \notin L_{\mathbf{X}_j}\right] + \Pr_{\mathbf{D}}[\mathsf{Iso}(\mathsf{addr}(\mathbf{D})) = 0]. \qquad (5)
\end{aligned}$$

Note that the first term is at most $t^2 \cdot 2^{-(n-(\log(t)+k+1))}$, since no $\mathbf{X}_j$ can assign $2^{-(\log(t)+k+1)}$ probability mass to more than $2^{\log(t)+k+1}$ points. To address the second term, let us consider the effect of applying addr to $\mathbf{D}$. Whenever any $\mathbf{D}_i$ satisfies $\mathsf{Iso}(\mathbf{D}_i) = 1$, we have $\mathsf{Iso}(\mathsf{addr}(\mathbf{D})) = \mathsf{Iso}(\mathbf{D}_i) = 1$. Otherwise, $\mathsf{Iso}(\mathbf{D}_i) = 0$ for all i, and we have $\mathsf{Iso}(\mathsf{addr}(\mathbf{D})) = \mathsf{Iso}(\mathbf{U}^n)$. Thus,

$$\begin{aligned}
\mathbf{Pr}[\mathsf{Iso}(\mathsf{addr}(\mathbf{D})) = 0] &= \Pr_{\mathbf{D}}\left[\mathsf{Iso}(\mathbf{U}^n) = 0 \text{ and } \bigcap_i (\mathsf{Iso}(\mathbf{D}_i) = 0)\right] \\
&= \mathbf{Pr}[\mathsf{Iso}(\mathbf{U}^n) = 0]^{t+1} \leq (1-\alpha)^{t+1}.
\end{aligned}$$

Plugging these two bounds back into (5) yields

$$\Pr_{\mathbf{D}}\left[\bigcap_i \left(\mathbf{D}_i \in \bigcap_j L_{\mathbf{X}_j}\right) \text{ and } \mathsf{Iso}(\mathsf{addr}(\mathbf{D})) = 1\right] \geq 1 - (1-\alpha)^{t+1} - t^2 \cdot 2^{-(n-(\log(t)+k+1))}.$$

Recalling the upper bound from (4), we conclude

$$\|\mathbf{X} - \mathbf{D}\|_{\mathsf{TV}} \geq 1 - (1-\alpha)^{t+1} - 2^{-(n-k)}\left(2t^3+1\right) - \beta. \qquad \blacktriangleleft$$

3 Preliminaries

We now briefly review some notation and formal definitions used throughout the work.

3.1 The Basics

For a positive integer n, we use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. For a binary string x, we use $|x|$ to denote the Hamming weight of x. All logarithms given in the paper are base 2. For two real numbers a and b, we write $a \pm b$ to denote a value in the range $[a-b, a+b]$. The indicator function is denoted by $\mathbf{1}(\cdot)$. The notation $\binom{n}{\leq k}$ is shorthand for $\sum_{i=0}^{k}\binom{n}{i}$. We use $\mathbb{F}_2$ to denote the finite field of two elements; we often identify it with $\{0,1\}$. The concatenation of two strings x and y is denoted $x \circ y$.

3.1.1 Asymptotics

We use the standard $\Omega(\cdot), O(\cdot), \Theta(\cdot)$ asymptotic notation to hide universal positive constants, although we will sometimes use $\gg$ and $\ll$ in more informal contexts. Occasionally, we will use subscripts to indicate an unspecified dependence on a particular parameter (e.g., $\Omega_d(n)$). Additionally, we write $o_t(1)$ to denote a positive quantity tending to 0 as t tends to infinity; we often omit the subscript when t is clear from context. The shorthand $\mathsf{poly}(n)$ corresponds to a polynomial in n of some fixed, but unspecified, degree.

3.1.2 Probability

We endeavor to use bold capital letters to denote probability distributions, and use bold lowercase letters to denote randomly drawn samples. That is, $\mathbf{x} \sim \mathbf{X}$ denotes a sample $\mathbf{x}$ drawn from the distribution $\mathbf{X}$. Oftentimes, we will refer to a distribution as a *source* if it is being fed into an extractor-like object. We reserve $\mathbf{U}$ for the uniform distribution over $\{0,1\}$. We use calligraphic letters, such as $\mathcal{X}$, for classes of distributions. For an event $\mathcal{E}$, we define $\mathbf{X}(\mathcal{E})$ to be the probability mass assigned to $\mathcal{E}$ by $\mathbf{X}$. For a function f, we use $f(\mathbf{X})$ to denote the output distribution of $f(\mathbf{x})$ on randomly drawn $\mathbf{x} \sim \mathbf{X}$. The *min-entropy* of a distribution $\mathbf{X}$, denoted $H_\infty(\mathbf{X})$, is given by $-\log \max_{x \in \mathsf{supp}(\mathbf{X})} \mathbf{Pr}[\mathbf{X} = x]$, where the support $\mathsf{supp}\,(\mathbf{X}) = \{x : \mathbf{Pr}[\mathbf{X} = x] > 0\}$.

Given a distribution $\mathbf{X}$ and positive integer t, we use $\mathbf{X}^t$ to denote the t-fold product distribution $\mathbf{X} \times \cdots \times \mathbf{X}$. If s is a finite set, we write $\mathbf{X}^s$ to emphasize that the coordinates of $\mathbf{X}^{|s|}$ are indexed by s. We refer to $\mathbf{X}$ as a mixture if it can be written as a convex combination of other distributions. That is, there exists $c_1, \ldots, c_k \in [0,1]$ and distributions $\mathbf{X}_1, \ldots, \mathbf{X}_k$ such that $\mathbf{X}(\mathcal{E}) = \sum_{i=1}^k c_i \cdot \mathbf{X}_i(\mathcal{E})$ for every event $\mathcal{E}$. Occasionally, we write this more concisely as $\mathbf{X} = \sum_{i=1}^k c_i \mathbf{X}_i$.

We measure the similarity of two (discrete) distributions $\mathbf{P}$ and $\mathbf{Q}$ by the *total variation (TV) distance*

$$\|\mathbf{P} - \mathbf{Q}\|_{\mathsf{TV}} = \max_{\text{event } \mathcal{E}} \mathbf{P}(\mathcal{E}) - \mathbf{Q}(\mathcal{E}) = \frac{1}{2} \sum_x |\mathbf{P}(x) - \mathbf{Q}(x)| \,.$$

We say $\mathbf{P}$ is ε-*close* to $\mathbf{Q}$ if $\|\mathbf{P} - \mathbf{Q}\|_{\mathsf{TV}} \leq \varepsilon$, and ε-*far* otherwise.

3.1.3 Explicit Constructions

The focus of this work is on *explicitly* constructing distributions with certain properties, by which we mean $\mathbf{D}$ can be sampled in $\mathsf{poly}(n)$ time. Note that as in the computational world, it is straightforward to non-constructively prove strong hardness results via a counting argument. More precisely, for any class $\mathcal{X}$ of distributions over $\{0,1\}^n$ of size $2^{2^{cn}}$ for a constant $c < 1$, there exists a uniform distribution $\mathbf{D}$ with distance $1 - 2^{-\Omega(n)}$ from every distribution in $\mathcal{X}$ (see the full version for details).

There is a small subtlety, however, in that the sampling models we consider have unbounded input length, so one cannot naively apply the above claim. Fortunately, the classes of interest can typically be approximated by a subset of the class where the input length is bounded (e.g., [14]), which allows the argument to go through. We are unaware of any work formally describing these counting arguments giving hard distributions for low-degree polynomial sources and $\mathsf{AC}^0[\oplus]$ sources, so we record the details in the appendix of the full version.

3.2 Extractors, Robust Extractors, and Isolators

There are three pseudorandom objects at the core of our work: extractors, robust extractors, and isolators. We state the formal definitions of the first two below.

▶ **Definition 10** ((Robust) Extractor). *A function* $\mathsf{Ext}\colon \{0,1\}^n \to \{0,1\}^m$ *is an* (ε, k)-extractor *for a class* $\mathcal{X}$ *of distributions over* $\{0,1\}^n$ *if for every source* $\mathbf{X} \in \mathcal{X}$ *with min-entropy at least* k*, we have*

$$\|\mathsf{Ext}(\mathbf{X}) - \mathbf{U}^m\|_{\mathsf{TV}} \leq \varepsilon.$$

If additionally, there exists some $z \in \{0,1\}^m$ *such that every source* $\mathbf{X} \in \mathcal{X}$ *satisfies*

$$\mathbf{Pr}[\mathbf{X} \in L \text{ and } \mathsf{Ext}(\mathbf{X}) = z] \leq \frac{1}{2^m} + \delta,$$

where $L = \{x \in \{0,1\}^n : \mathbf{Pr}[\mathbf{X} = x] \leq 2^{-k}\}$*, then we call* Ext *an* (ε, δ, k)-robust extractor.

It is worth mentioning that there are other ways to formalize this notion of robustness. For example, one could ask for an extractor which works for a source $\mathbf{X}$, as long as $\mathbf{X}$ has TV distance no more than some parameter γ to a high min-entropy source $\mathbf{X}'$ (i.e., an extractor for sources of high smooth min-entropy [38]). Assuming such extractors existed for the classes of distributions we consider, it would be possible to obtain a constant distance lower bound as in the second part of Theorem 8.

Unfortunately, it does not seem feasible to get distance approaching 1. Fix some choice of γ and a sampler $f(\mathbf{U}^r)$ for the hard distribution $\mathbf{D}$ considered in Theorem 8. If $f(\mathbf{U}^r)$ is γ-far from $\mathbf{D}$, we trivially have a lower bound of γ, so assume this is not the case. Note that we cannot expect a "smooth extractor" to have a better upper bound probability guarantee than $2^{-m} + \gamma$, since it has to apply to distributions which are a point mass with probability γ and uniform otherwise. Tracing through the remainder of the argument, one finds that $f(\mathbf{U}^r)$ has distance roughly $1 - \varepsilon - \gamma$ from $\mathbf{D}$ for some small ε. In other words, we can only guarantee a bound around $\min(\gamma, 1 - \varepsilon - \gamma) \ll 1$.

One of the benefits of Definition 10 is that we can obtain tighter distance guarantees, because we do not need to consider the behavior of the extractor on heavy points. Hence, we can avoid the above issue. Returning back to our selected formalizations, we restate the definition of an isolator for the reader's convenience.

▶ **Definition 6** (Isolator). *A function* $\mathsf{Iso}\colon \{0,1\}^n \to \{0,1\}$ *is an* (α, β, k)-isolator *for a class* $\mathcal{X}$ *of distributions over* $\{0,1\}^n$ *if*

1. $\mathbf{Pr}[\mathsf{Iso}(\mathbf{U}^n) = 1] \geq \alpha$*, and*
2. *Every source* $\mathbf{X} \in \mathcal{X}$ *satisfies* $\mathbf{Pr}_{x \sim \mathbf{X}}[x \in L \text{ and } \mathsf{Iso}(x) = 1] \leq \beta$*, where* $L = \{x \in \{0,1\}^n : \mathbf{Pr}[\mathbf{X} = x] \leq 2^{-k}\}$.

The role of robust extractors in the present work is primarily as a convenient device for constructing isolators.

▶ **Fact 11.** *Let* $\mathcal{X}$ *be a class of distributions over* $\{0,1\}^n$ *which includes the uniform distribution* $\mathbf{U}^n$*. If* $\mathsf{rExt}\colon \{0,1\}^n \to \{0,1\}^m$ *is an* (ε, δ, k)*-robust extractor for* $\mathcal{X}$*, then the function* $\mathsf{Iso}(x) \coloneqq \mathbf{1}(\mathsf{rExt}(x) = z)$ *is a* $(2^{-m} - \varepsilon, 2^{-m} + \delta, k)$*-isolator for* $\mathcal{X}$ *(where* z *is the same string as in Definition 10).*

Of course, one still has to construct a robust extractor to apply Fact 11. Typically, this is not much more difficult than constructing a traditional extractor, and in the following section, we illustrate such a modification with a number of natural examples. We note that in certain cases, the robustness can even be obtained for free.

▷ Claim 12. Let $\mathcal{X}$ be a class of distributions over $\{0,1\}^n$ such that each $\mathbf{X} \in \mathcal{X}$ is the uniform distribution on some set $S_{\mathbf{X}} \subseteq \{0,1\}^n$ (i.e., $\mathcal{X}$ is a family of *flat* sources). If $\mathsf{Ext}\colon \{0,1\}^n \to \{0,1\}^m$ is an (ε, k)-extractor for $\mathcal{X}$, then Ext is also an $(\varepsilon, \varepsilon, k)$-robust extractor for $\mathcal{X}$.

Proof. Consider any source $\mathbf{X} \in \mathcal{X}$, which is uniform over a set $S_{\mathbf{X}}$. If $|S_{\mathbf{X}}| \geq 2^k$, then $\mathbf{X}$ has min-entropy at least k and the set

$$L \coloneqq \{x \in \{0,1\}^n : \mathbf{Pr}[\mathbf{X} = x] \leq 2^{-k}\} = \{0,1\}^n.$$

Thus, the extractor guarantee implies

$$\mathop{\mathbf{Pr}}_{x\sim\mathbf{X}}[x \in L \text{ and } \mathsf{Ext}(x) = 0^m] = \mathop{\mathbf{Pr}}_{x\sim\mathbf{X}}[\mathsf{Ext}(x) = 0^m] \leq 2^{-m} + \varepsilon.$$

Otherwise $|S_{\mathbf{X}}| < 2^k$, and $\mathbf{Pr}[\mathbf{X} = x] = 0$ for every $x \in L$. Hence,

$$\mathop{\mathbf{Pr}}_{x\sim\mathbf{X}}[x \in L \text{ and } \mathsf{Ext}(x) = 0^m] \leq \mathop{\mathbf{Pr}}_{x\sim\mathbf{X}}[x \in L] = 0. \qquad \triangleleft$$

4 Hard-to-Sample Distributions for Polynomial Sources

In this section, we instantiate Theorem 8 for polynomial sources.

▶ **Definition 13** (Polynomial Source). *A degree-d* polynomial source $\mathbf{X}$ *is defined by a polynomial map $P\colon \mathbb{F}_2^m \to \mathbb{F}_2^n$, where $P = (p_1, p_2, \ldots, p_n)$ and each p_i is an $\mathbb{F}_2$-polynomial of degree at most d, such that $\mathbf{X} = P(\mathbf{U}^m)$.*

Prior to our work, the best explicit distribution was only known to have distance $2^{-\Omega(n)}$ from any low-degree polynomial map; this is achieved by combining an argument of [46] (described in Subsection 2.1) with an extractor construction of [13]. We improve this distance bound to $1 - o(1)$.

▶ **Theorem 14.** *There exists a constant $\delta > 0$ such that the following holds. Let N and Δ be positive integers satisfying $\Delta \leq \delta \log\log N$. There exist positive integers n, t, d with $(n+1)t \leq N$ and an isolator $\mathsf{Iso}\colon \{0,1\}^n \to \{0,1\}$ with suitable parameters for the class of polynomial sources on $\{0,1\}^n$ of degree d such that the following holds.*

Define the distribution

$$\mathbf{D} = (\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_t, \mathsf{Iso}(\mathbf{U}_1), \mathsf{Iso}(\mathbf{U}_2), \ldots, \mathsf{Iso}(\mathbf{U}_t)),$$

where $\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_t$ are independent copies of $\mathbf{U}^n$. Let $\mathcal{X}$ be the class of all polynomial sources $\mathbf{X}$ on $\{0,1\}^{(n+1)t}$ of degree Δ. Then for any $\mathbf{X} \in \mathcal{X}$,

$$\|\mathbf{D} - \mathbf{X}\|_{\mathsf{TV}} \geq 1 - O\left(\frac{\Delta}{\log\log N} \log\left(\frac{\log\log N}{\Delta}\right)\right).$$

Moreover, the distribution $\mathbf{D}$ can be sampled in $\mathsf{poly}(N)$ time.

Additionally, there exists an isolator $\widetilde{\mathsf{Iso}}\colon \{0,1\}^n \to \{0,1\}$ (with possibly different parameters) such that the distribution $\widetilde{\mathbf{D}} = (\mathbf{U}^n, \widetilde{\mathsf{Iso}}(\mathbf{U}^n))$ satisfies $\left\|\widetilde{\mathbf{D}} - \mathbf{Y}\right\|_{\mathsf{TV}} \geq 1/4 - n^{-\Omega(1)}$ for all degree-Δ polynomial sources $\mathbf{Y}$ on $\{0,1\}^{n+1}$. $\widetilde{\mathbf{D}}$, too, can be sampled in $\mathsf{poly}(N)$ time.

We prove Theorem 14 by modifying the construction of an explicit extractor from [13] to create an isolator Iso for low-degree polynomial sources, and invoke Theorem 8 using Iso.

The extractor from [13] is constructed by brute-forcing over many possible extractors on a small number of input bits. The existence of such an extractor is guaranteed by the probabilistic method (as is standard), but what makes their argument not straightforward is that the number of polynomial sources is not bounded, since a polynomial source can have any number of input bits. Their key ingredient is an input reduction technique [13, Theorem 4.1], which shows that it is enough to consider only sources where the number of input bits is at most a constant factor times the min-entropy.

We observe below that a variant of their input reduction argument also works for isolators. We first prove a variant of their simple entropy smoothing claim [13, Claim 4.3].

▷ Claim 15. For any random variable $\mathbf{X}$ over $\{0,1\}^n$ and positive integer k, there exists a function $S\colon \{0,1\}^n \to \{0,1\}^{k+1}$ with the following property. For every $x \in \{0,1\}^n$ such that $\mathbf{Pr}[\mathbf{X} = x] \le 2^{-k}$, we have $\mathbf{Pr}[S(\mathbf{X}) = S(x)] \le 2^{-k}$.

Proof. We perform the following merging operation. Start with each string in $\{0,1\}^n$ in its own bucket. If there are two buckets whose combined probability mass under $\mathbf{X}$ is at most 2^{-k}, then merge the two buckets. This operation is repeated for as long as possible. (This process necessarily terminates since the number of buckets decreases in each step.)

At the end, there can be at most one bucket whose corresponding probability is at most $2^{-(k+1)}$, since otherwise we could merge two such buckets. This implies that there are at most 2^{k+1} buckets in total, and we can define S to simply map each string $x \in \{0,1\}^n$ to the bucket containing it (where we associate each bucket with a distinct string in $\{0,1\}^{k+1}$).

To verify the desired property of S, first observe that since any bucket containing two or more elements is the result of some merge operation, it must have total probability at most 2^{-k}. This handles all x such that $|S(x)| \ge 2$. On the other hand, if $\mathbf{Pr}[\mathbf{X} = x] \le 2^{-k}$ and $|S(x)| = 1$, then clearly $\mathbf{Pr}[S(\mathbf{X}) = S(x)] = \mathbf{Pr}[\mathbf{X} = x] \le 2^{-k}$. ◁

We also need the following general lemma, which is implicit [13]. We sketch the proof for completeness.

▶ **Lemma 16.** *Let $f\colon \mathbb{F}_2^r \to \mathbb{F}_2^n$ be a function and $0 < \varepsilon < 1/4$. Let $\ell = \lceil n + 3\log(1/\varepsilon) \rceil$. If $r > \ell$, there exist $A \in \mathbb{F}_2^{r \times \ell}$ and $b \in \mathbb{F}_2^r$ such that if we define $h\colon \mathbb{F}_2^\ell \to \mathbb{F}_2^n$ by $h(x) = f(Ax+b)$, we have $\left\| f(\mathbf{U}^r) - h(\mathbf{U}^\ell) \right\|_{\mathsf{TV}} \le 2\varepsilon$.*

Proof. Lemma 4.2 in [13] gives a full rank[6] linear map $M\colon \mathbb{F}_2^r \to \mathbb{F}_2^{r-\ell}$ such that

$$\left\| f(\mathbf{U}^r) \circ M(\mathbf{U}^r) - f(\mathbf{U}^r) \circ \mathbf{U}^{r-\ell} \right\|_{\mathsf{TV}} \le 2\varepsilon. \tag{6}$$

We start by observing that

$$\left\| f(\mathbf{U}^r) \circ M(\mathbf{U}^r) - f(\mathbf{U}^r) \circ \mathbf{U}^{r-\ell} \right\|_{\mathsf{TV}} = \mathop{\mathbb{E}}_{v \sim M(\mathbf{U}^r)} \Big[\left\| (f(\mathbf{U}^r) \,|\, M(\mathbf{U}^r) = v) - f(\mathbf{U}^r) \right\|_{\mathsf{TV}} \Big],$$

where we have used that M has full rank, so $M(\mathbf{U}^r)$ is $\mathbf{U}^{r-\ell}$. Combining with (6), there must exist some $v \in \mathbb{F}_2^{r-\ell}$ such that

$$\left\| (f(\mathbf{U}^r) \,|\, M(\mathbf{U}^r) = v) - f(\mathbf{U}^r) \right\|_{\mathsf{TV}} \le 2\varepsilon.$$

Now $(f(\mathbf{U}^r) \,|\, M(\mathbf{U}^r) = v)$ can be expressed as $h(\mathbf{U}^\ell)$ for a function h on ℓ inputs of the desired form $h(x) = f(Ax + b)$, since conditioning on $M(\mathbf{U}^r) = v$ (where M has full rank) is equivalent to replacing $r - \ell$ of the r input bits for f by affine functions of the other ℓ inputs. Thus, we have $\left\| h(\mathbf{U}^\ell) - f(\mathbf{U}^r) \right\|_{\mathsf{TV}} \le 2\varepsilon$ as desired. ◀

[6] The conference version of [13] does not explicit state that M is full rank, but this can be found in the updated arXiv version: `https://arxiv.org/abs/2309.11019`.

We now proceed to the input reduction lemma.

▶ **Lemma 17.** *Suppose* $\mathsf{Iso}\colon \mathbb{F}_2^n \to \mathbb{F}_2$ *is an* $(\alpha, \beta, k-1)$*-isolator for the class of degree-*d *polynomial sources with at most* $4(k+1)$ *inputs. Then* Iso *is also an* $(\alpha, \beta + 2^{-k}, k)$*-isolator for the class of all degree-*d *polynomial sources.*

Proof. We clearly have $\mathbf{Pr}[\mathsf{Iso}(\mathbf{U}^n) = 1] \geq \alpha$ since Iso is an $(\alpha, \beta, k-1)$-isolator, so it remains to verify that Iso satisfies the second condition in the definition of an $(\alpha, \beta + 2^{-k}, k)$-isolator for the class of all degree-d polynomial sources.

Let $f\colon \mathbb{F}_2^r \to \mathbb{F}_2^n$ be a degree-d polynomial map with light points $L_f \coloneqq \{x : \mathbf{Pr}[f(\mathbf{U}^r) = x] \leq 1/2^k\}$. Let $\ell = 4(k+1)$, and let $S\colon \{0,1\}^n \to \{0,1\}^{k+1}$ be the map given by Claim 15 applied to $f(\mathbf{U}^r)$ and k. Apply Lemma 16 to the function $S(f(\cdot))$ with $\varepsilon = 2^{-(k+1)}$ to get $A \in \mathbb{F}_2^{r \times \ell}, b \in \mathbb{F}_2^r$ such that

$$\left\| S(f(A \cdot \mathbf{U}^\ell + b)) - S(f(\mathbf{U}^r)) \right\|_{\mathsf{TV}} \leq 2^{-k}. \tag{7}$$

Let $g\colon \mathbb{F}_2^\ell \to \mathbb{F}_2^n$ be defined by $g(x) = f(Ax + b)$. Observe that g is a degree-d polynomial since we have only substituted linear polynomials for the inputs of f.

Define $L_g = \{x : \mathbf{Pr}[g(\mathbf{U}^\ell) = x] \leq 2^{-(k-1)}\}$. Observe that $L_f \subseteq L_g$, since (7) implies every $x \in L_f$ satisfies

$$\mathbf{Pr}[g(\mathbf{U}^\ell) = x] \leq \mathbf{Pr}[S(g(\mathbf{U}^\ell)) = S(x)] \leq \mathbf{Pr}[S(f(\mathbf{U}^r)) = S(x)] + 2^{-k} \leq 2^{-(k-1)}.$$

Therefore, we conclude

$$\begin{aligned}
\mathop{\mathbf{Pr}}_{\mathbf{x} \sim f(\mathbf{U}^r)}[\mathbf{x} \in L_f \text{ and } \mathsf{Iso}(\mathbf{x}) = 1] &\leq \mathop{\mathbf{Pr}}_{\mathbf{x} \sim g(\mathbf{U}^\ell)}[\mathbf{x} \in L_f \text{ and } \mathsf{Iso}(\mathbf{x}) = 1] + 2^{-k} && \text{(by (7))} \\
&\leq \mathop{\mathbf{Pr}}_{\mathbf{x} \sim g(\mathbf{U}^\ell)}[\mathbf{x} \in L_g \text{ and } \mathsf{Iso}(\mathbf{x}) = 1] + 2^{-k} && \text{(since } L_f \subseteq L_g\text{)} \\
&\leq \beta + 2^{-k}.
\end{aligned}$$

The last inequality above uses the isolator guarantee for $g(\mathbf{U}^\ell)$. This finishes the proof. ◀

To apply Lemma 17, we need to construct an isolator for the class of polynomial sources with bounded input length. The following lemma will allow us to find such an object more efficiently than a naive brute force. Below, recall that a family $\{f_i\colon \{0,1\}^n \to \{0,1\}^m\}_i$ of t-wise uniform hash functions is defined by the property that for any string $(y_1, y_2, \ldots, y_t) \in (\{0,1\}^m)^t$ and all distinct strings $x_1, x_2, \ldots, x_t \in \{0,1\}^n$, a function $\mathbf{f}$ chosen uniformly at random from the family satisfies $(\mathbf{f}(x_1), \mathbf{f}(x_2), \ldots, \mathbf{f}(x_t)) = (y_1, y_2, \ldots, y_t)$ with probability 2^{-mt}.

▶ **Lemma 18.** *Let* $0 < \alpha < \beta < 1$*. Let* k, n, ℓ*, and* d *be positive integers. Set* $K = 2^k$ *and* $N = 2^n$*. Let* $t \geq 4$ *be an even integer. Suppose there exists an integer* m *such that* $p \coloneqq 2^{-m}$ *satisfies* $\alpha < p < \beta$ *and the following inequality holds:*

$$\left(\frac{Npt + t^2}{N^2(p-\alpha)^2}\right)^{t/2} + 2^{\binom{\ell}{\leq d}n}\left(\frac{Kpt + t^2}{K^2(\beta - p)^2}\right)^{t/2} < 1/8.$$

Additionally, let $\mathcal{H}$ *be a family of* t*-wise uniform hash functions from* $\{0,1\}^n$ *to* $\{0,1\}^m$*. Then there exists a function* $h \in \mathcal{H}$ *such that the function* g *defined by* $g(x) = \mathbf{1}(h(x) = 0^m)$ *is an* (α, β, k)*-isolator for the class of degree-*d *polynomial sources with* ℓ *inputs and* n *outputs.*

Proof. Let $\mathbf{h}$ be a function drawn uniformly at random from $\mathcal{H}$, and define $\mathbf{g}\colon \{0,1\}^n \to \{0,1\}$ by $\mathbf{g}(x) = \mathbf{1}(\mathbf{h}(x) = 0^m)$. By the assumption on $\mathcal{H}$, $\{\mathbf{g}(x)\}_{x\in\{0,1\}^n}$ is a collection of t-wise independent random variables, and for each x, $\mathbf{Pr}[\mathbf{g}(x) = 1] = 2^{-m}$. By using a tail inequality for sums of t-wise independent random variables (see [9, Lemma 2.3]), we have

$$\mathop{\mathbf{Pr}}_{\mathbf{g}}[\mathbf{Pr}[\mathbf{g}(\mathbf{U}^n) = 1] \le \alpha] \le 8\left(\frac{Npt + t^2}{N^2(p-\alpha)^2}\right)^{t/2}.$$

This is the probability that $\mathbf{g}$ fails to satisfy the first condition for being an (α, β, k)-isolator.

We will now estimate the probability that for some source, the second condition for being an (α, β, k)-isolator is not satisfied. Fix any source $\mathbf{X}$ on $\{0,1\}^n$, and define $L = \{x : \mathbf{Pr}[\mathbf{X} = x] \le 2^{-k}\}$. We wish to show that with high probability over $\mathbf{g}$, $\mathbf{Pr}[\mathbf{g}(\mathbf{X}) = 1 \text{ and } \mathbf{X} \in L] \le \beta$. Let $\mathbf{Z} = \mathbf{Pr}[\mathbf{g}(\mathbf{X}) = 1 \text{ and } \mathbf{X} \in L]$ be the random variable of interest. For each $x \in L$, define the indicator random variable $\mathbf{Z}_x = \mathbf{1}(\mathbf{g}(x) = 1)$, each of which is a Bernoulli random variable with probability p of being 1. Then $\mathbf{Z} = \sum_{x\in L} \mathbf{Pr}[\mathbf{X} = x]\mathbf{Z}_x$ and $\mathbb{E}[\mathbf{Z}] = \mathbf{Pr}[\mathbf{X} \in L] \cdot p \le p$.

By a tail inequality for t-wise independent random variables as above and using that $\mathbf{Pr}[\mathbf{X} = x] \le 2^{-k}$ for all $x \in L$, we obtain

$$\begin{aligned}\mathbf{Pr}[\mathbf{Z} \ge \beta] &= \mathbf{Pr}[\mathbf{Z} \ge \mathbb{E}[\mathbf{Z}] + \beta - \mathbf{Pr}[\mathbf{X} \in L] \cdot p] \\ &\le \mathbf{Pr}[\mathbf{Z} \ge \mathbb{E}[\mathbf{Z}] + \beta - p] \\ &\le 8\left(\frac{Kpt + t^2}{K^2(\beta-p)^2}\right)^{t/2}.\end{aligned}$$

Now a union bound over the $2^{\binom{\ell}{\le d}n}$ many degree-d polynomial sources with ℓ inputs and n outputs gives that the second condition for being an (α, β, k)-isolator for such sources does not hold with probability at most $8 \cdot 2^{\binom{\ell}{\le d}n} \cdot \left(\frac{Kpt+t^2}{K^2(\beta-p)^2}\right)^{t/2}$. Combining this with the failure probability for the first condition shows that g fails to be an (α, β, k)-isolator for such sources with probability at most

$$8\left(\left(\frac{Npt + t^2}{N^2(\alpha-p)^2}\right)^{t/2} + 2^{\binom{\ell}{\le d}n}\left(\frac{Kpt + t^2}{K^2(\beta-p)^2}\right)^{t/2}\right) < 1$$

by assumption. Hence there exists some $\mathbf{g}$ which is an (α, β, k)-isolator, as desired. ◀

By iterating through a t-wise uniform family of hash functions, we must find an isolator in a reasonable amount of time.

▶ **Corollary 19.** *Let k, n, d, and m be positive integers. Suppose $k \ge 10(d + \log n)$, $k \le n$, and $m \le 0.01k$. For $p = 2^{-m}$, there exist $\alpha = p - 2^{-\Omega(n)}$ and $\beta = p + 2^{-\Omega(k)}$ such that there exists an (α, β, k)-isolator* Iso *for the class of degree-d polynomials with n outputs, which can be computed in time $2^{O(\binom{\Theta(k)}{\le d}n^2)}$.*

Proof. Set $\ell = 5k$, $K = 2^k$, and $t = 2\left(\binom{\ell}{\le d}n + 4\right)$; note that $K^{0.99} \ge 4t$. By using Lemma 18 under the conditions on k, d in the statement, there exists a function Iso which is an $(\alpha', \beta', k-1)$-isolator for the class of degree-d polynomial sources with $4(k+1) \le 5k$ inputs and n outputs, where $\alpha' = p - 2^{-\Omega(n)}$ and $\beta' = p + 2^{-\Omega(k)}$. Furthermore by Lemma 17, Iso is an (α, β, k)-isolator for the class of degree-d polynomial sources with n outputs, where $\alpha = \alpha' = p - 2^{-\Omega(n)}$ and $\beta = \beta' + 2^{-k} = p + 2^{-\Omega(k)}$.

We compute such a function by going over any fixed t-wise uniform family of hash functions from $\{0,1\}^n$ to $\{0,1\}^m$ until we find a function that lets us construct an isolator as described in Lemma 18. It is well known that there is such a family $\mathcal{H}$ of size 2^{tn} where each

function in the family can be evaluated in $\mathsf{poly}(n, m, t)$ time (see, for instance, [43, Corollary 3.34]). For any such fixed function $h \in \mathcal{H}$, we check whether g defined by $g(x) = \mathbf{1}(h(x) = 0^m)$ is an (α, β, k)-isolator. This can be done in time $2^{O(\binom{\ell}{\leq d} n)}$. ◀

By invoking Theorem 8 with the above isolator, we obtain an explicit hard distribution for polynomial sources, proving Theorem 14. We restate the theorem below for the reader's convenience.

▶ **Theorem 14.** *There exists a constant $\delta > 0$ such that the following holds. Let N and Δ be positive integers satisfying $\Delta \leq \delta \log\log N$. There exist positive integers n, t, d with $(n+1)t \leq N$ and an isolator $\mathsf{Iso}\colon \{0,1\}^n \to \{0,1\}$ with suitable parameters for the class of polynomial sources on $\{0,1\}^n$ of degree d such that the following holds.*

Define the distribution

$$\mathbf{D} = (\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_t, \mathsf{Iso}(\mathbf{U}_1), \mathsf{Iso}(\mathbf{U}_2), \ldots, \mathsf{Iso}(\mathbf{U}_t)),$$

where $\mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_t$ are independent copies of $\mathbf{U}^n$. Let $\mathcal{X}$ be the class of all polynomial sources $\mathbf{X}$ on $\{0,1\}^{(n+1)t}$ of degree Δ. Then for any $\mathbf{X} \in \mathcal{X}$,

$$\|\mathbf{D} - \mathbf{X}\|_{\mathsf{TV}} \geq 1 - O\left(\frac{\Delta}{\log\log N} \log\left(\frac{\log\log N}{\Delta}\right)\right).$$

Moreover, the distribution $\mathbf{D}$ can be sampled in $\mathsf{poly}(N)$ time.

Additionally, there exists an isolator $\widetilde{\mathsf{Iso}}\colon \{0,1\}^n \to \{0,1\}$ (with possibly different parameters) such that the distribution $\widetilde{\mathbf{D}} = (\mathbf{U}^n, \widetilde{\mathsf{Iso}}(\mathbf{U}^n))$ satisfies $\left\|\widetilde{\mathbf{D}} - \mathbf{Y}\right\|_{\mathsf{TV}} \geq 1/4 - n^{-\Omega(1)}$ for all degree-Δ polynomial sources $\mathbf{Y}$ on $\{0,1\}^{n+1}$. $\widetilde{\mathbf{D}}$, too, can be sampled in $\mathsf{poly}(N)$ time.

Proof. We choose parameters to optimize the final distance with respect to the distribution length $(n+1)t$ while ensuring the construction takes only $\mathsf{poly}(N)$ time. Set $n = \lfloor (\log N)^{1/(2+\lambda)} \rfloor$ for a constant $\lambda > 0$ to be determined later. Let $d = \log n$ and $t = \lfloor d/\Delta \rfloor - 1$ so that $\Delta(t+1) \leq d$. By picking δ to be small enough, we have $t \geq 1$. Let $k = \lceil 20 \log n \rceil$ so that the condition $k \geq 10(\log n + d)$ in Corollary 19 is satisfied. Let m be the largest integer such that for $p = 2^{-m}$, we have $t \geq \frac{1}{p} \log \frac{1}{p}$. Since $t \leq d = \log n$ and $m \leq \log t$, we have $m \leq 0.01k$. Now let Iso be the (α, β, k)-isolator given by Corollary 19 with the chosen parameters and $\alpha = p - 2^{-\Omega(n)}, \beta = p + 2^{-\Omega(k)}$.

Let $\mathbf{X} \in \mathcal{X}$. Note that $\mathsf{addr}_{n,t}(\mathbf{X})$ can be computed by a degree $\Delta(t+1) \leq d$ polynomial source since $\mathsf{addr}_{n,t}$ can be computed by a degree-$(t+1)$ polynomial. By Theorem 8, we have

$$\begin{aligned}
1 - \|\mathbf{D} - \mathbf{X}\|_{\mathsf{TV}} &\leq (1-\alpha)^t + \beta + (2t^3+1)2^{-(n-k)} \\
&\leq (1 - p + 2^{-\Omega(n)})^t + p + 2^{-\Omega(k)} + (2t^3+1) \cdot 2^{-\Omega(n)} \\
&\leq (1-p)^t + t \cdot 2^{-\Omega(n)} + p + 2^{-\Omega(k)} + 2^{-\Omega(n)} \\
&\leq \exp(-pt) + p + n^{-\Omega(1)} \\
&\leq O(p) \leq O\left(\frac{\log t}{t}\right) \\
&\leq O\left(\frac{\Delta}{\log\log N} \log\left(\frac{\log\log N}{\Delta}\right)\right).
\end{aligned}$$

Finally, we show that $\mathbf{D}$ can be sampled in time $\mathsf{poly}(N)$. The isolator Iso can be computed in time $2^{O(\binom{\Theta(k)}{\leq d} n^2)}$. We have

$$\binom{\Theta(k)}{\leq d} \leq \left(\frac{e \cdot \Theta(k)}{d}\right)^d \leq C_0^{\log n} = n^{\log C_0}$$

for some constant $C_0 > 1$. Set $\lambda = \log C_0$, so Iso can be computed in time $2^{O(n^{2+\lambda})} = \mathsf{poly}(N)$. Given that Iso is computable in $\mathsf{poly}(N)$ time, it is straightforward to see that $\mathbf{D}$ can be sampled in $\mathsf{poly}(N)$ time.

We now specialize to the case of $t = 1$ and $\widetilde{\mathbf{D}} = (\mathbf{U}^n, \widetilde{\mathsf{Iso}}(\mathbf{U}^n))$, where $\widetilde{\mathsf{Iso}}$ is an $(1/2 - 2^{-\Omega(n)}, 1/2 + n^{-\Omega(1)}, 20\log n)$-isolator. Invoking Theorem 8, we obtain

$$\left\|\widetilde{\mathbf{D}} - \mathbf{Y}\right\|_{\mathsf{TV}} \geq 1 - (1/2 + 2^{-\Omega(n)})^2 - (1/2 + n^{-\Omega(1)}) - 2^{-\Omega(n)} \geq 1/4 - n^{-\Omega(1)}. \quad \blacktriangleleft$$

Note that our hard distribution above has at most N bits instead of exactly N bits. To obtain a distribution with exactly N bits, one can pad with some bits that are fixed to, say, 0. It is clear that the padded distribution does not become easier for polynomial sources of degree Δ since TV distance cannot increase by applying a projection. It is also easy to see that the padded distribution is computable in $\mathsf{poly}(N)$ time if the original distribution is computable in $\mathsf{poly}(N)$ time.

5 Open Problems

We conclude by highlighting several directions for future research. One notable goal is to bring our understanding of sampling by small circuits with parity gates (i.e., $\mathsf{AC}^0[\oplus]$) up to the frontier of what is known in the computational setting.

1. Provide an explicit distribution over $\{0,1\}^n$ which has distance $1 - o(1)$ from the output of any $\mathsf{AC}^0[\oplus]$ circuit with n outputs, $\mathsf{poly}(n)$ many gates, and arbitrarily many random input bits.

Even obtaining a much weaker bound would be novel; to the best of our knowledge, no hardness results for explicit[7] distributions exist beyond the trivial arguments of finding a hard distribution for extremely small circuits via brute force, or appealing to the limitations posed by binary precision. For example, the $(1/3)$-biased distribution over $\{0,1\}$ cannot be exactly generated by a circuit with r random bits as input, since each output occurs with probability that is an integer multiple of 2^{-r}. Of course, one could also ask for a similarly hard distribution in the case of circuits with mod p gates for any other prime p. Without these more powerful gates, it is known that such circuits cannot accurately generate the uniform distribution over the codewords of a good code [35, 8]. Unfortunately, the arguments do not seem to generalize.

In light of Theorem 14, a reasonable approach to Question 1 is to construct an isolator for $\mathbb{F}_2$-polynomial sources of polylogarithmic degree, and then appeal to classical results on polynomial approximations of circuits [37, 41]. This is a more general degree constraint than the best-known explicit extractors can handle [13], but we are optimistic further improvements in these extractors will be flexible enough for our techniques to apply. It is also worth emphasizing that extractors and isolators are incomparable objects, and it is plausible that the latter may be easier to construct in certain settings.

An alternative strengthening of Theorem 14 is to improve the quantitative behavior; while we can prove a $1 - o(1)$ bound, the specific decay rate of the $o(1)$ term is suboptimal.

2. Provide an explicit distribution over $\{0,1\}^n$ which has distance $1 - \exp(-n^{\Omega_d(1)})$ from the output of any degree-d $\mathbb{F}_2$-polynomial source.

[7] Existential results can be found in the appendix of the full version.

We note that one can prove such a bound in the case of $d = 1$, where the source $P(\mathbf{U}^r)$ is uniform over an affine subspace. Letting $\mathsf{Ext}\colon \{0,1\}^n \to \{0,1\}^m$ be an (ε, k)-extractor for affine sources, we can set the hard distribution $\mathbf{D}$ to be uniform over the preimage $\mathsf{Ext}^{-1}(0^m)$. If $P(\mathbf{U}^r)$ has min-entropy at least k, then it has distance at least $1 - 2^{-m} - \varepsilon$ from $\mathbf{D}$. Otherwise, $P(\mathbf{U}^r)$ is uniform over a set of size less than 2^k, so it has distance at least $1 - 2^{-(n-k)}/(2^{-m} - \varepsilon)$ from $\mathbf{D}$. By known explicit constructions of such objects [12, 51, 34], we can take $k = \Omega(n)$, $\varepsilon = 2^{-\Omega(n)}$, and $m = \Omega(n)$ with a sufficiently small implicit constant to conclude $\|P(\mathbf{U}^r) - \mathbf{D}\|_{\mathsf{TV}} \geq 1 - \exp(-\Omega(n))$.

Sadly, this argument already breaks down for $d = 2$, since we no longer get meaningful information about $\mathsf{supp}\,(P(\mathbf{U}^r))$ in the low min-entropy case. Moreover, our current framework does not appear strong enough to address Question 2, and it would be interesting to determine whether the framework itself can be strengthened.

3. Prove a quantitatively stronger version of Theorem 8.

More specifically, it would be desirable to have the β term in Theorem 8 decrease as t increases. It also seems worthwhile to try to improve the $t = 1$ setting to be able to obtain the optimal form $\frac{1}{2} - o(1)$, rather than our current $\frac{1}{4} - o(1)$. In both cases, one may wish to consider a variant of extractors even beyond robust extractors or isolators.

References

1 Miklós Ajtai. Σ_1^1-formulae on finite structures. *Ann. Pure Appl. Logic*, 24(1):1–48, 1983. doi:10.1016/0168-0072(83)90038-6.

2 Yaroslav Alekseev, Mika Göös, Konstantin Myasnikov, Artur Riazanov, and Dmitry Sokolov. Sampling permutations with cell probes is hard. In *Proceedings of the 58th Annual ACM Symposium on Theory of Computing*, 2026.

3 Omar Alrabiah, Jesse Goodman, Jonathan Mosheiff, and João Ribeiro. Low-degree polynomials are good extractors. In *Approximation, randomization, and combinatorial optimization. Algorithms and techniques*, volume 353 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 38, 25. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/lipics.approx/random.2025.38.

4 Andris Ambainis, Leonard J. Schulman, Amnon Ta-Shma, Umesh Vazirani, and Avi Wigderson. The quantum communication complexity of sampling. *SIAM J. Comput.*, 32(6):1570–1585, 2003. doi:10.1137/S009753979935476.

5 Anurag Anshu, Yangjing Dong, Fengning Ou, and Penghui Yao. On the computational power of QAC0 with barely superlinear ancillae. In *STOC'25—Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 1476–1487. ACM, New York, 2025. doi:10.1145/3717823.3718189.

6 László Babai. Random oracles separate PSPACE from the polynomial-time hierarchy. *Information Processing Letters*, 26(1):51–53, 1987. doi:10.1016/0020-0190(87)90036-6.

7 Marshall Ball, Ronen Shaltiel, and Jad Silbak. Extractors for samplable distributions with low min-entropy. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 596–603, 2025. doi:10.1145/3717823.3718143.

8 Chris Beck, Russell Impagliazzo, and Shachar Lovett. Large deviation bounds for decision trees and sampling lower bounds for AC0-circuits. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 101–110. IEEE, 2012. doi:10.1109/FOCS.2012.82.

9 Mihir Bellare and John Rompel. Randomness-efficient oblivious sampling. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 276–287. IEEE, 1994. doi:10.1109/SFCS.1994.365687.

10 Adam Bene Watts and Natalie Parham. Unconditional quantum advantage for sampling with shallow circuits. In *17th Innovations in Theoretical Computer Science Conference*, volume 362 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 17, 12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. doi:10.4230/lipics.itcs.2026.17.

11 Ravi B Boppana and Jeffrey C Lagarias. One-way functions and circuit complexity. *Information and Computation*, 74(3):226–240, 1987. doi:10.1016/0890-5401(87)90022-8.

12 Jean Bourgain. On the construction of affine extractors. *Geom. Funct. Anal.*, 17(1):33–57, 2007. doi:10.1007/s00039-007-0593-z.

13 Eshan Chattopadhyay, Jesse Goodman, and Mohit Gurumukhani. Extractors for polynomial sources over $\mathbb{F}_2$. In *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*, volume 287 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ITCS.2024.28.

14 Eshan Chattopadhyay, Jesse Goodman, and David Zuckerman. The space complexity of sampling. In *13th Innovations in Theoretical Computer Science Conference (ITCS)*, 2022.

15 Anindya De and Thomas Watson. Extractors and lower bounds for locally samplable sources. *ACM Transactions on Computation Theory (TOCT)*, 4(1):1–21, 2012. doi:10.1145/2141938.2141941.

16 Harm Derksen, Peter Ivanov, Chin Ho Lee, and Emanuele Viola. Pseudorandomness, symmetry, smoothing: I. In *39th Computational Complexity Conference*, volume 300 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 18, 27. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/lipics.ccc.2024.18.

17 Dean Doron and William M. Hoza. Implications of better PRGs for permutation branching programs. In *Approximation, randomization, and combinatorial optimization. Algorithms and techniques*, volume 353 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 28, 20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/lipics.approx/random.2025.28.

18 Stephen Fenner, Daniel Grier, Daniel Padé, and Thomas Thierauf. Tight bounds on depth-2 QAC-circuits computing parity. *arXiv preprint*, 2025. doi:10.48550/arXiv.2504.06433.

19 Merrick Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Math. Systems Theory*, 17(1):13–27, 1984. doi:10.1007/BF01744431.

20 Alexander Golovnev, Zeyu Guo, Pooya Hatami, Satyajeet Nagargoje, and Chao Yan. Hilbert functions and low-degree randomness extractors. In *Approximation, randomization, and combinatorial optimization. Algorithms and techniques*, volume 317 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 41, 24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/lipics.approx/random.2024.41.

21 Mika Göös and Thomas Watson. A lower bound for sampling disjoint sets. *ACM Trans. Comput. Theory*, 12(3):Art. 20, 13, 2020. doi:10.1145/3404858.

22 Daniel Grier, Daniel M. Kane, Jackson Morris, Anthony Ostuni, and Kewen Wu. Quantum advantage from sampling shallow circuits: beyond hardness of marginals. In *17th Innovations in Theoretical Computer Science Conference*, volume 362 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 73, 14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. doi:10.4230/lipics.itcs.2026.73.

23 Daniel Grier, Jackson Morris, and Kewen Wu. QAC^0 contains TC^0 (with many copies of the input). *arXiv preprint*, 2026. arXiv:2601.03243.

24 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 6–20, 1986. doi:10.1145/12130.12132.

25 Johan Håstad. *Computational limitations for small depth circuits*. PhD thesis, Massachusetts Institute of Technology, 1986.

26 Pooya Hatami and William Hoza. Theory of unconditional pseudorandom generators. *Foundations and Trends in Theoretical Computer Science*, 16(1-2):1–210, February 2024. doi:10.1561/0400000109.

27 William M. Hoza and Zelin Lv. On sums of INW pseudorandom generators. In *Approximation, randomization, and combinatorial optimization. Algorithms and techniques*, volume 353 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 67, 24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/lipics.approx/random.2025.67.

28 Malvika Raj Joshi, Avishay Tal, Francisca Vasconcelos, and John Wright. Improved lower bounds for QAC^0. *arXiv preprint*, 2025. arXiv:2512.14643.

29 Daniel M Kane, Anthony Ostuni, and Kewen Wu. Locality bounds for sampling Hamming slices. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1279–1286, 2024. doi:10.1145/3618260.3649670.

30 Daniel M. Kane, Anthony Ostuni, and Kewen Wu. Symmetric distributions from shallow circuits. *arXiv preprint*, 2025. doi:10.48550/arXiv.2511.14127.

31 Mohammad Mahdi Khodabandeh and Igor Shinkar. On sampling lower bounds for polynomials, 2026. Available at https://eccc.weizmann.ac.il/report/2026/066/.

32 Vinayak M. Kumar. New pseudorandom generators and correlation bounds using extractors. In *16th Innovations in Theoretical Computer Science Conference*, volume 325 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 68, 23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/lipics.itcs.2025.68.

33 Chin Ho Lee and Emanuele Viola. Pseudorandom bits for non-commutative programs. In *40th Computational Complexity Conference*, volume 339 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 9, 22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/lipics.ccc.2025.9.

34 Xin Li. A new approach to affine extractors and dispersers. In *26th Annual IEEE Conference on Computational Complexity*, pages 137–147. IEEE Computer Soc., Los Alamitos, CA, 2011. doi:10.1109/CCC.2011.27.

35 Shachar Lovett and Emanuele Viola. Bounded-depth circuits cannot sample good codes. In *2011 IEEE 26th Annual Conference on Computational Complexity*, pages 243–251. IEEE, 2011. doi:10.1109/CCC.2011.11.

36 Shivam Nadimpalli, Natalie Parham, Francisca Vasconcelos, and Henry Yuen. On the Pauli spectrum of QAC0. In *STOC'24—Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1498–1506. ACM, New York, 2024. doi:10.1145/3618260.3649662.

37 Alexander A Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical Notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987.

38 R. Renner and S. Wolf. Smooth Renyi entropy and applications. In *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.*, pages 233–, 2004. doi:10.1109/ISIT.2004.1365269.

39 Ronen Shaltiel. Multiplicative extractors for samplable distributions. In *40th Computational Complexity Conference*, volume 339 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 22, 22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/lipics.ccc.2025.22.

40 Ronen Shaltiel and Jad Silbak. Explicit codes for poly-size circuits and functions that are hard to sample on low entropy distributions. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 2028–2038, 2024. doi:10.1145/3618260.3649735.

41 Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 77–82, 1987. doi:10.1145/28395.28404.

42 Luca Trevisan and Salil Vadhan. Extracting randomness from samplable distributions. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 32–42. IEEE, 2000. doi:10.1109/SFCS.2000.892063.

43 Salil P Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.

44 Emanuele Viola. The complexity of distributions. *SIAM Journal on Computing*, 41(1):191–218, 2012. doi:10.1137/100814998.

45 Emanuele Viola. Extractors for Turing-machine sources. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 663–671. Springer, 2012. doi:10.1007/978-3-642-32512-0_56.

46 Emanuele Viola. Extractors for circuit sources. *SIAM Journal on Computing*, 43(2):655–672, 2014. doi:10.1137/11085983X.

47 Emanuele Viola. Quadratic maps are hard to sample. *ACM Transactions on Computation Theory (TOCT)*, 8(4):1–4, 2016. doi:10.1145/2934308.

48 Emanuele Viola. Sampling lower bounds: boolean average-case and permutations. *SIAM Journal on Computing*, 49(1):119–137, 2020. doi:10.1137/18M1198405.

49 Emanuele Viola. New Sampling Lower Bounds via the Separator. In *38th Computational Complexity Conference (CCC 2023)*, volume 264 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 26:1–26:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.CCC.2023.26.

50 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles. In *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, pages 1–10. IEEE, 1985.

51 Amir Yehudayoff. Affine extractors over prime fields. *Combinatorica*, 31(2):245–256, 2011. doi:10.1007/s00493-011-2604-9.

52 Huacheng Yu and Wei Zhan. Sampling, Flowers and Communication. In *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*, volume 287 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 100:1–100:11. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ITCS.2024.100.

Almost All Graphs Are Vertex-Minor Universal

Ruben Ascoli
Department of Mathematics, Georgia Institute of Technology, Atlanta, GA, USA

Bryce Frederickson
Department of Mathematics, Emory University, Atlanta, GA, USA

Sarah Frederickson
Department of Mathematics, Georgia Institute of Technology, Atlanta, GA, USA

Caleb McFarland
Department of Mathematics, Georgia Institute of Technology, Atlanta, GA, USA

Logan Post
Department of Mathematics, Georgia Institute of Technology, Atlanta, GA, USA

Abstract

Answering a question of Claudet, we prove that the uniformly random graph $G \sim \mathbb{G}(n, 1/2)$ is $\Omega(\sqrt{n})$-vertex-minor universal with high probability. That is, for some constant $\alpha \approx 0.911$, any graph on any $\alpha\sqrt{n}$ specified vertices of G can be obtained as a vertex-minor of G. This has direct implications for quantum communications networks: an n-vertex k-vertex-minor universal graph corresponds to an n-qubit k-stabilizer universal graph state, which has the property that one can induce any stabilizer state on any k qubits using only local operations and classical communications.

We further employ our methods in two other contexts. We obtain a bipartite pivot-minor version of our main result, and we use it to derive a universality statement for minors in random binary matroids. We also introduce the *vertex-minor Ramsey number* $R_{\mathrm{vm}}(k)$ to be the smallest value n such that every n-vertex graph contains an independent set of size k as a vertex-minor. Supported by our main result, we conjecture that $R_{\mathrm{vm}}(k)$ is polynomial in k. We prove $\Omega(k^2) \leq R_{\mathrm{vm}}(k) \leq 2^k - 1$.

2012 ACM Subject Classification Mathematics of computing → Random graphs; Theory of computation → Quantum computation theory

Keywords and phrases vertex-minors, random graphs, quantum networks, graph states

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.38

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2602.09049` [1]

Funding *Caleb McFarland*: Supported in part by the National Science Foundation under Grant No. DMS- 2452111.

Acknowledgements This work began at the Fall 2025 Atlanta Graduate Combinatorics Workshop at Georgia Tech. We thank Georgia Tech for providing funding for the workshop, Sean Longbrake for helping to organize it, and Elliot Frederickson for his motivating presence there. We also extend our thanks to Xiaoyu He and Rose McCarty for enlightening conversations, and to Rose McCarty for telling us about the problem of whether the uniformly random graph is vertex-minor universal (which was originally posed by Nathan Claudet) and pointing us to relevant sources. Finally, we thank Nathan Claudet for helpful comments on an earlier version of this paper.

© Ruben Ascoli, Bryce Frederickson, Sarah Frederickson, Caleb McFarland, and Logan Post;
licensed under Creative Commons License CC-BY 4.0

1 Background and main results

Vertex-minors have been the star of numerous fascinating problems in graph theory since the 1980s. They often serve as a dense analogue of the usual minor relation, and so similarly have many deep applications to width parameters, algorithms, and logic [30, 22, 15, 5]. For more background on vertex-minors and their applications, see the recent survey on vertex-minors by Kim and Oum [28]. We begin with their definition.

▶ **Definition 1.** *Given a graph G and a vertex $v \in V(G)$, a* local complementation *at v is an operation which replaces $G[N(v)]$, the subgraph of G induced on the neighborhood of v, by its complement. The resulting graph is denoted $G * v$ and has the same vertex set as G. A graph H with $V(H) \subseteq V(G)$ is a* vertex-minor *of G if it can be obtained from G through a sequence of local complementations and vertex deletions.*

Vertex-minors also have fascinating connections to quantum information theory, on which we elaborate in Section 1.1. The following definition, introduced by Claudet et al. [12] and further studied by Cautrès et al. [8], provides a bridge between the purely graph-theoretic concept of vertex-minors and the quantum-theoretic notion of k-stabilizer universal states.

▶ **Definition 2.** *A graph G is* k-vertex-minor universal *on $V(G)$ if for every vertex subset $U \subseteq V(G)$ with $|U| = k$, every graph H on U is a vertex-minor of G.*

We emphasize that we require each H to be obtained exactly on each specified vertex set U. In particular, there are $\binom{|V(G)|}{k}2^{\binom{k}{2}}$ total choices of pairs U and H.

Our main result, resolving a question of Claudet [11], is that for some constant C, *almost all* graphs on Ck^2 vertices are k-vertex-minor universal. That is, a uniformly random graph on Ck^2 vertices is k-vertex-minor universal with probability tending exponentially fast to 1 as k increases. The quadratic dependence is best possible up to the constant factor C [12]. We use the notation $\mathbb{G}(n,p)$ for the Erdős-Renyi distribution on graphs on n vertices where each edge is included independently with probability p; in particular, $\mathbb{G}(n,1/2)$ is the uniform distribution on graphs on n vertices.

▶ **Theorem 3.** *For all $c > 0$, if $G \sim \mathbb{G}(n,1/2)$ with $n \geq (1+c)\frac{1}{2\log_2(4/3)}k^2$, then G is k-vertex-minor universal with probability at least $1 - 2^{-(1+o(1))ck^2/2}$.*

On the practical side, our explicit bound on the probability of success converges exponentially quickly beyond the threshold value. Thus we have a probabilistic algorithm to produce a k-vertex-minor universal graph on $O(k^2)$ vertices. Explicitly, letting $C = \frac{1}{2\log_2(4/3)} \approx 1.205$, we have that for any $\varepsilon > 0$ and $k \geq 3$, a uniformly random graph on $n = \lceil Ck^2 + 16k\log k + 4\log(1/\varepsilon)\rceil$ vertices is k-vertex-minor universal with probability at least $1-\varepsilon$. This follows from keeping track of the error terms in the union bound (see (3)).

The proof of Theorem 3 is non-constructive, and we do not obtain an algorithm for finding a desired vertex-minor in $\mathbb{G}(n,1/2)$. We note that this problem is NP-hard in general [17, 16], but this does not preclude the possibility of an efficient algorithm which succeeds with high probability when the input is a random graph.

▶ **Question.** *Does there exist a polynomial-time algorithm which takes as input $G \sim \mathbb{G}(n,1/2)$ and a graph H on $O(\sqrt{n})$ vertices, and with high probability outputs a sequence of local complementations verifying that H is a vertex-minor of G?*

For comparison to known results regarding k-vertex-minor universal graphs, Claudet et al. [12] showed that there exist k-vertex-minor universal graphs with $O(k^4 \log k)$ vertices, while any k-vertex-minor universal graph must have at least $(k-2)^2/(2\log_2 3)$ vertices. Cautrès et al. [8] improved the existence result to match the lower bound up to a constant factor, establishing the existence of k-vertex-minor universal graphs on $(2+o(1))k^2$ vertices. Their construction is based on a random unbalanced bipartite graph G where one side has $\Theta(k^2)$ vertices and the other has $\Theta(k \log k)$ vertices, with edges between the two sides included independently with probability $1/2$. This yields a graph with $\Theta(k^3 \log k)$ edges with high probability, which is less than a constant proportion of possible edges. They also give a polynomial time algorithm to find a k-vertex graph H as a vertex-minor in G with high probability.

Our Theorem 3 improves the constant factor 2 from Cautrès et al. [8] to $\frac{1}{2\log_2(4/3)} \approx 1.205$, progressing closer to the lower bound [12] of $1/(2\log_2 3) \approx 0.315$. More conceptually important, however, is that our result states that almost all graphs – including dense and non-bipartite graphs – also have the vertex-minor universality property. This is perhaps surprising in the context of a paper of Gross, Flammia, and Eisert [24], which shows that all but an exponentially small fraction of quantum states are too entangled to be computationally useful. Our theorem, in contrast, shows that all but an exponentially small fraction of $O(k^2)$-qubit *graph* states are in fact k-stabilizer universal. We elaborate more on the quantum perspective in Section 1.1.

Although the vertex-minor problem addressed by Theorem 3 is most relevant to quantum information theory, our methods extend to a stronger notion of universality. The following definition plays a key role.

▶ **Definition 4.** *Given a graph G and an edge $uv \in E(G)$, a pivot at the edge uv is the operation which replaces G by $G \times uv := G * u * v * u$. It is easy to verify that $G*u*v*u = G*v*u*v$, so the operation is well-defined. A graph H with $V(H) \subseteq V(G)$ is a* pivot-minor *of G if it can be obtained from G through a sequence of pivots and vertex deletions. A graph G is* k-pivot-minor universal *on $V(G)$ if for every vertex subset $U \subseteq V(G)$ with $|U| = k$, every graph H on U is a pivot-minor of G.*

Our techniques naturally extend to pivot-minors. Pivot-minors in bipartite graphs are essentially equivalent to minors in binary matroids, and thus we also obtain the following two results. For brevity, we omit proofs of these results and precise definitions for matroid minor universality, but we include them in the full version of this manuscript [1].

▶ **Theorem 5.** *For all $c > 0$, if $G \sim \mathbb{G}(n, 1/2)$ with $n \geq (1+c)\frac{2}{\log_2(16/13)}k^2 \approx (1+c)6.68k^2$, then G is k-pivot-minor universal with probability at least $1 - 2^{-(1+o(1))ck^2}$.*

▶ **Theorem 6.** *Let $C = \frac{1}{2\log_2(8/7)} + \frac{1}{4} \approx 2.85$. For all $c > 0$, if M is a uniformly random binary matroid on ground set $[n]$ and $n \geq (1+c)2Ck^2$, then M is k-minor universal with probability at least $1 - 2^{-(1+o(1))ck^2/4}$.*

For G to be k-vertex-minor universal requires that we can obtain every graph on every k-subset of vertices of G as a vertex-minor. We can also ask a related question: how big does n need to be to ensure that *every* graph on n vertices contains a *particular* graph H as a vertex-minor on some k vertices? The answer is only finite if the graph H has no edges, or is an *independent set*; indeed, otherwise, a graph of any size n with no edges does not contain H as a vertex-minor.

▶ **Definition 7.** *Let $R_{\mathrm{vm}}(k)$ be the minimum n such that any n-vertex graph contains the independent set of size k as a vertex-minor.*

Recall the classical Ramsey number $R(s,t)$ which is the smallest n such that any n-vertex graph contains either an independent set of size s or a clique of size t. As observed in [6], we have $R_{\mathrm{vm}}(k) \leq R(k,k+1)$, since if a graph contains a clique of size $k+1$, locally complementing at one vertex forms an independent set on the remaining k vertices. The current best upper bound on $R(k,k+1)$ is roughly 3.8^k, due to Gupta, Ndiaye, Norin, and Wei [25] following the breakthrough work of Campos, Griffiths, Morris, and Sahasrabudhe [7]. Here we provide a nontrivial, but still exponential upper bound for $R_{\mathrm{vm}}(k)$, as well as a quadratic lower bound obtained from the random graph.

▶ **Theorem 8.** *For all $k \geq 0$,*

$$(1+o(1))\frac{1}{2\log_2 3}k^2 \leq R_{\mathrm{vm}}(k) \leq 2^k - 1.$$

The lower bound comes from showing that with high probability, the uniformly random graph $G \sim \mathbb{G}(n,1/2)$ does not contain an independent set of size k as a vertex-minor if $k \geq (1+\varepsilon)\sqrt{2\log_2(3)n}$, for any $\varepsilon > 0$. Due to Theorem 3, this is best possible up to the constant factor, and we believe the lower bound is closer to the true answer.

▶ **Conjecture 9.** $R_{\mathrm{vm}}(k) \leq \mathsf{poly}(k)$.

Paper organization. We conclude this section by discussing the connection between vertex-minor universality and quantum information theory. Then, in Section 2, we provide the technical graph-theoretic and probabilistic tools we need for the proof of Theorem 3, state some preliminary lemmas, and give a detailed overview of the proof. In Section 3, we carry out our proof strategy for Theorem 3. Finally, in Section 4, we prove Theorem 8 about the vertex-minor Ramsey number and conclude with some remaining open questions.

1.1 Motivation from quantum information theory

Although the study of k-vertex-minor universal graphs has only taken off in recent years, the relationship between quantum graph states and vertex-minors dates back to 2004 with the pioneering work of Van den Nest, Dehaene, and De Moor [31]. In the years since, many authors have studied various models of random graph states and their algorithmic and quantum entanglement properties; see e.g. [32, 16, 27, 23] and the references therein for non-exhaustive coverage of the subject.

Recent work in quantum information theory has emphasized the role of entangled multipartite states as reusable resources for large-scale quantum communication networks. In such networks, separate parties are restricted to Local Operations and Classical Communication (LOCC), with no ability to perform joint quantum operations once the resource state has been distributed. A central question in this setting is to understand which global entangled states allow pairs of parties to generate entangled EPR pairs.

Motivated by this perspective, Bravyi, Sharma, Szegedy, and de Wolf [4] introduced the notion of k-pairable quantum states. An n-party state $|\psi\rangle$ is k-pairable if, for any choice of k disjoint pairs of parties, there exists a LOCC protocol that starts with $|\psi\rangle$ and ends in a

state such that each of those pairs of parties share an EPR pair. This definition reflects a worst-case notion of network universality: a single fixed resource state must, under LOCC, generate EPR pairs for any choice of k disjoint pairs of parties. Bravyi et al. [4] established upper and lower bounds on the achievable value of k in terms of the number of parties and number of qubits per party.

Subsequently, Claudet, Mhalla, and Perdrix [12] and Cautrès, Claudet, Mhalla, Perdrix, Savin, and Thomassé [8] elaborated on the ideas of Bravyi et al. [4], focusing on the case where each party has just one qubit and studying a generalization of k-pairable states which Cautrés et al. [8] call *k-stabilizer universal* states. A state is k-stabilizer universal if any stabilizer state on any subset of k qubits can be obtained through LOCC protocols. Stabilizer universality is a stronger notion than k-pairability: EPR pairs are stabilizer states, so any $2k$-stabilizer universal state is also k-pairable. For more background on stabilizer states, see for example [26].

The quantum-theoretic contributions of [12] and [8], as well as those of this article, rely on the connection between quantum states and graph theory. Vertex-minors capture an important sequence of operations on quantum states: if H can be obtained as a vertex-minor of G, then the quantum graph state corresponding to H can be obtained from the graph state corresponding to G through just local Clifford operations, local Pauli measurements, and classical communications. (The converse also holds if H has no isolated vertices [18].) It follows that if G is a k-vertex-minor universal graph then the quantum graph state corresponding to G is k-stabilizer universal. This enables explicit constructions of universal resource states for quantum communication protocols [12, 8].

2 Preliminaries and proof overview

We denote the symmetric difference of sets by $A \triangle B = (A \setminus B) \cup (B \setminus A)$. For a graph G, we use $V(G)$ and $E(G)$ to denote its vertex set and edge set respectively. For $U \subseteq V(G)$, we use $G[U]$ to denote the graph induced on U, and $G - U$ to denote the graph obtained by deleting U. The *adjacency matrix* of a graph G (over $\mathbb{F}_2$) is the symmetric matrix $A \in \mathbb{F}_2^{V(G)\times V(G)}$ such that $A_{u,v} = 1$ if $uv \in E(G)$ and $A_{u,v} = 0$ otherwise. Note that as we assume graphs to be simple, A has 0 along the diagonal. For disjoint sets X, Y, we define the set $[X, Y] \coloneqq \{\{x, y\} \mid x \in X, y \in Y\}$ and let $G[X, Y]$ be the induced bipartite subgraph with edge set $E(G) \cap [X, Y]$.

2.1 Vertex-minor preliminaries

Throughout the paper it will be convenient to have notation which represents both the local complementation and pivot operations. To that end we define the following. Let $x \in \{\varnothing, \{v\}, \{u, v\}\}$. Then

$$G \circ x = \begin{cases} G & x = \varnothing \\ G * v & x = \{v\} \\ G \times uv & x = \{u, v\}. \end{cases}$$

Implicitly we assume that if $x = \{u, v\}$, then $uv \in E(G)$, otherwise the operation is not defined. If $(x_i)_{i=1}^{\ell}$ is a sequence of such x, then we often use $G \circ (x_i)_{i=1}^{\ell}$ as shorthand for $G \circ x_1 \circ x_2 \circ \cdots \circ x_n$. We similarly write $G * (v_i)_{i=1}^{\ell}$ as shorthand for $G * v_1 * \cdots * v_\ell$.

Observe that an equivalent way to view the pivot operation is that for $vu \in E(G)$, $G \times vu$ is the graph obtained from G by taking the symmetric difference of G with the complete tripartite graph with parts $N(v) \cap N(u)$, $N(v) \setminus (N(u) \cup \{u\})$, $N(u) \setminus (N(v) \cup \{v\})$, and then swapping the labels of u and v.

We now discuss some well-known facts about vertex-minors and pivot-minors. We say two graphs are *locally equivalent* if one can be obtained from the other by a sequence of local complementations. Similarly we say that two graphs are *pivot equivalent* if one can be obtained from the other by a sequence of pivots. It is easy to see that if H is a vertex-(pivot-) minor of G, then $H \subseteq G'$ for some graph G' which is locally (pivot) equivalent to G. The following fact is well-known.

▶ **Lemma 10** ([30, Proposition 2.5])**.** *If $vu \in E(G)$, then $uw \in E(G \times vu)$ if and only if $vw \in E(G)$. Furthermore, if both edges exist, we have*

$$G \times vu \times uw = G \times vw.$$

The following result, proven independently by Bouchet [3] and Fon-Der-Flaass [21], will be crucial in reordering the operations when finding a vertex-minor.

▶ **Theorem 11** ([3, 21])**.** *Suppose H is a vertex-minor of G and $v \in V(G) \setminus V(H)$. Then H is a vertex-minor of*

1. *$G - v$,*
2. *$G * v - v$, or*
3. *$G \times vu - v$ for some $u \in N(v)$.*

We note that Lemma 10 implies that the choice of u does not matter up to pivot-equivalence. That is, if H is a vertex-minor of $G \times vu - v$ for some $u \in N(v)$, then H is a vertex-minor of $G \times vu' - v$ for every $u' \in N(v)$. This helps us prove our key structural lemma below. We call this our "reordering lemma," because it allows us to reorganize a sequence of local complementations in a convenient way.

▶ **Lemma 12.** *Fix a set $\hat{V}$. Let $v_1, \dots, v_\ell \in \hat{V}$ be a (possibly repeating) sequence and let $\hat{G}$ be a graph with $V(\hat{G}) = \hat{V}$. Then there is a sequence $x_1, \dots, x_r$ of singletons and pairs from $\hat{V}$ such that for any graph G with $\hat{V} \subseteq V(G)$ and $G[\hat{V}] = \hat{G}$, we have*

- **(I)** *$G * v_1 * v_2 \cdots * v_\ell - \hat{V} = G \circ x_1 \cdots \circ x_r - \hat{V}$.*
- **(II)** *The sets $x_1, \dots, x_r$ are pairwise disjoint.*
- **(III)** *If $v \in \hat{V}$ appears exactly once in $(v_i)_{i=1}^{\ell}$, then $v \in x_j$ for some $j \in [r]$.*

Moreover, any choice of $(x_i)_i$ satisfying (I) and (II) for all G also satisfies (III).

We emphasize that the sequence $(x_i)_i$ depends *only* on the subgraph $\hat{G} = G[\hat{V}]$, and is independent of the edges between $\hat{V}$ and $V(G) \setminus \hat{V}$. Without this careful tracking of the dependency, this lemma would be implied by known results, (see [14, Proposition 6] or [13, Proposition 7]). In Section 3.3, we give a short proof by induction using Theorem 11.

Because of the dependence only on $\hat{G}$, if $G \sim \mathbb{G}(n, 1/2)$, then by revealing only the edges of $\hat{G}$ we can rewrite any arbitrary sequence of local complementations $*(v_i)_i$ in the desired form $\circ(x_j)_j$, and then study the statistics of the resulting graph.

2.2 Probability preliminaries

We use $\mathbb{G}(n,p)$ to denote the Erdős-Rényi random graph distribution on vertex set $[n]$, where each edge is present independently with probability p. We let $\mathcal{G}_n$ denote the set of all labeled graphs on vertex set $[n]$. The following simple proposition is crucial for our proof.

▶ **Proposition 13.** *If $G \sim \mathbb{G}(n,1/2)$ and $v \in [n]$, then $G * v \sim \mathbb{G}(n,1/2)$.*

Proof. Since the map $G \mapsto G*v$ is an involution on $\mathcal{G}_n$, it preserves the uniform distribution. Indeed, for any graph $G' \in \mathcal{G}_n$, we have $\mathbb{P}[G*v = G'] = \mathbb{P}[G = G'*v] = \frac{1}{|\mathcal{G}_n|}$. ◀

The basic probabilistic tool we rely on is Chebyshev's inequality, which we use in the following form.

▶ **Lemma 14** (Chebyshev's inequality). *For a random variable X and $\alpha > 0$, we have*

$$\mathbb{P}[|X - \mathbb{E}[X]| \geq \alpha] \leq \frac{\mathrm{Var}[X]}{\alpha^2}.$$

In particular, setting $\alpha = \mathbb{E}[X]$ gives $\mathbb{P}[X = 0] \leq \frac{\mathrm{Var}[X]}{\mathbb{E}[X]^2}$.

2.3 Proof overview

Before the complete proof in Section 3, we outline the argument for Theorem 3. Let $G \sim \mathbb{G}(n+k,1/2)$. Fix a vertex set $U \subseteq V(G)$ of size $|U| = k$, and fix a graph H on U. Denote the vertices of G by $V = \{v_1, \ldots, v_n\} \sqcup U$. To form a copy of H on vertex set U, we locally complement on some subset of the vertices in $\{v_1, \ldots, v_n\}$, given by index set J. Complementing on this subset J produces a random graph on U, which we denote by $G_J[U]$. If two choices for the subset J are sufficiently different, then the resulting graphs $G_J[U]$ will likely also be different. There are 2^n choices for J and $2^{\binom{k}{2}}$ choices for the graph H. Hence for $k = \Theta(\sqrt{n})$, we are likely to find a choice of J for which the graph $G_J[U]$ is equal to H. The core difficulty of the proof is analyzing the difference, or more precisely the correlation, between G_J and $G_{J'}$ for different subsets J, J'.

We now proceed with a more technical overview of our proof. For each subset $J \subseteq [n]$, where $J = \{i_1 < i_2 < \ldots < i_{|J|}\}$ we define

$$G_J := G * v_{i_1} * v_{i_2} * \cdots * v_{i_{|J|}}, \tag{1}$$

so we can track 2^n distinct procedures for locally complementing on G. For example, $G_\emptyset = G$ and $G_{[n]} = G * v_1 * \cdots * v_n$. Let X be the number of "successful" procedures, i.e. $X = \sum_{J \subseteq [n]} X_J$ where X_J the indicator that $G_J[U] = H$. We want to show that $X > 0$, and hence H is a vertex-minor of G, with high probability. By Proposition 13, for each J we have $G_J \sim \mathbb{G}(n,1/2)$, so each X_J equals 1 with probability $2^{-\binom{k}{2}}$. Thus $\mathbb{E}[X] = 2^n 2^{-\binom{k}{2}} \to \infty$, and we upper bound $\mathbb{P}[X = 0]$ using Chebyshev's inequality. Finally, we apply a union bound over all choices of U and H to show that G is k-vertex-minor universal.

The central challenge for this argument is to control $\mathrm{Var}[X]$, and therefore to show that the various $G_J, G_{J'}$ are not too correlated. For example, comparing $G_\emptyset = G$ and $G_{[n]}$, we find that each time we locally complement at a vertex v_i, a random subset of the edges in U are flipped. Hence, we expect that after sufficiently many flips, the distributions of $G[U]$ and $G_{[n]}[U]$ will be almost entirely uncorrelated.

To make this precise, we define two random walks on the space of graphs with vertex set U (see Section 3.1). The first random walk chooses a uniformly random subset of U and complements the edges with both ends in that subset. This corresponds to locally complementing at a vertex outside of U that has a random neighborhood. The second random walk chooses two uniformly random subsets $S, T \subseteq U$ and toggles adjacency between pairs of vertices which lie in two different sets among $S \setminus T, T \setminus S$, and $S \cap T$. This corresponds to pivoting on an edge outside U where each end has a random neighborhood in U.

We view these random walks as linear transformations over the space of all graphs, and we compute their eigenvectors and eigenvalues exactly (see Lemma 16). We use this spectral calculation to conclude that both random walks quickly approach the uniform distribution (Lemma 20).

We then use these random walks to study $G_J[U] \triangle G_{J'}[U]$ when J and J' differ. Specifically, we argue that if $|J \triangle J'|$ is large, then $G_J[U] \triangle G_{J'}[U]$ is close to uniformly random, and thus the covariance between X_J and $X_{J'}$ is very small (see Lemma 24). Suppose $J = \{i_1 < i_2 < \cdots < i_{|J|}\}$ and $J' = \{j_1 < j_2 < \cdots < j_{|J'|}\}$. Because local complementation is an involution, we have

$$G_{J'} = G_J * v_{i_{|J|}} * \cdots * v_{i_1} * v_{j_1} * \cdots * v_{j_{|J'|}},$$

meaning that $G_{J'}[U]$ is formed from $G_J[U]$ by locally complementing on vertices outside of U. This sounds like our random walk, but as stated, the same vertex could be locally complemented on multiple times, and thus the steps in the corresponding random walk may be correlated. To combat this, we apply our reordering Lemma 12 to obtain a sequence $(x_i)_{i=1}^{\ell}$ where

$$G_{J'}[U] = G_J \circ (x_i)_{i=1}^{\ell}[U]$$

such that the x_i are pairwise disjoint. Then we may consider $G_J[U] \triangle (G_J \circ (x_i)_{i=1}^{t})[U]$ as a random walk starting at $t = 0$ with the empty graph and ending at $t = \ell$ with $G_J[U] \triangle G_{J'}[U]$. Property (III) of Lemma 12 implies that $|\bigcup_{i=1}^{\ell} x_i| \geq |J \triangle J'|$, so if $|J \triangle J'|$ is sufficiently large, then this random walk mixed for many steps, implying that $G_J[U] \triangle G_{J'}[U]$ is approximately uniformly random. Since typical sets J, J' differ in $n/2$ many indices, this gives the desired correlation bounds, implying that the variance of X is small and hence the probability that H is a vertex-minor of G is large (Lemma 26).

We note some subtlety in the sets of edges which must stay independent in this approach. In order to apply Lemma 12, we first reveal the edges of $G \setminus U$. Then we reveal the edges between x_i and U one at a time so that at each step, each vertex in x_i has a uniformly random neighborhood in U. We use Lemma 21 to argue that conditional on all previous steps, the unexposed edges remain uniformly random throughout the process. This guarantees that each step of the random walk is independent of the previous steps, and so each step brings us closer to the uniform distribution.

3 Proof of vertex-minor universality of the random graph

3.1 Random walks on the space of graphs

We first define and consider two random walks Com and Piv on the space of graphs with vertex set $U = [k]$. We begin with a graph Γ drawn from some starting distribution μ_0 on all graphs with vertex set U. Then at each time t, the Com walk simulates locally complementing

(and deleting) at vertex v outside of U with a uniformly random neighborhood. Similarly, the Piv walk simulates pivoting on an edge outside U with uniformly random neighborhoods. Formally, we define the random walks as follows.

▶ **Definition 15.** *Suppose μ is a probability distribution on $\mathcal{G}_k$, the set of labeled graphs with vertex set $[k]$. Identify μ with a vector in $[0,1]^{\mathcal{G}_k}$, and we write $\mu(H)$ to be the probability of the graph $H \in \mathcal{G}_k$. First, let $\mathrm{Com}(\mu)$ be the distribution of the graph $\Gamma \triangle K_S$, where $\Gamma \sim \mu$, and $S \subseteq [k]$ is a uniformly random subset independent of Γ. Next, for $S, T \subseteq [k]$ we let $K^{\triangle}_{S,T}$ be the complete tripartite graph with tripartition $\{S \setminus T, T \setminus S, S \cap T\}$. Then let $\mathrm{Piv}(\mu)$ be the distribution of $\Gamma \triangle K^{\triangle}_{S,T}$ where $\Gamma \sim \mu$ and $S, T \subseteq [k]$ are uniformly random and independent of Γ. Finally, let μ_{unif} be the uniform distribution on $\mathcal{G}_k$.*

It is easy to see that the uniform distribution μ_{unif} is stationary under both the Com and Piv operators, and we will be interested in how fast the distribution μ_0 approaches μ_{unif} under these operators. It is helpful to think of these as steps in a random walk on a Cayley graph of the additive group $\mathbb{F}_2^{E(K_k)}$. For example, if we construct the Cayley graph with multiset of generators $\{\mathbb{1}_{E(K_S)} : S \subseteq [k]\}$, then one step in this walk gives the convolution $\mu \mapsto \mathrm{Com}(\mu)$. A nice survey of random walks on groups is given by Diaconis [20]. As symmetric random walks, both $\mathrm{Com}, \mathrm{Piv} : \mathbb{R}^{\mathcal{G}_n} \to \mathbb{R}^{\mathcal{G}_n}$ are symmetric linear transformations over $\mathbb{R}$. Because they are walks on a Cayley graph, we can explicitly compute their eigenvectors.

▶ **Lemma 16.** *The eigenvectors of* Com *and* Piv *are given by $\{\chi_G \mid G \in \mathcal{G}_k\}$ where $\chi_G \in \mathbb{R}^{\mathcal{G}_k}$ is defined as $\chi_G(H) = (-1)^{|E(G) \cap E(H)|}$. For each $G \in \mathcal{G}_k$, let $\lambda_G^{\mathrm{Com}}, \lambda_G^{\mathrm{Piv}}$ denote the eigenvalues of χ_G under* Com, Piv *respectively. Then we have*

$$\lambda_G^{\mathrm{Com}} = \mathbb{E}_S\left[(-1)^{|E(G)\cap E(K_S)|}\right]$$
$$\lambda_G^{\mathrm{Piv}} = \mathbb{E}_{S,T}\left[(-1)^{|E(G)\cap E(K^{\triangle}_{S,T})|}\right]$$

where the expectation is over $S, T \subseteq [k]$ drawn uniformly at random.

Proof. The statement for Com follows from a simple calculation:

$$\begin{aligned}
\mathrm{Com}(\chi_G)(H) &= 2^{-k} \sum_{S \subseteq [k]} \chi_G(H \triangle K_S) \\
&= 2^{-k} \sum_{S \subseteq [k]} (-1)^{|E(G)\cap E(H)|}(-1)^{|E(G)\cap E(K_S)|} \\
&= \chi_G(H) \cdot 2^{-k} \sum_{S \subseteq [k]} (-1)^{|E(G)\cap E(K_S)|} \\
&= \chi_G(H) \cdot \mathbb{E}_S\left[(-1)^{|E(G)\cap E(K_S)|}\right].
\end{aligned}$$

The proof for Piv is similar. ◀

We can now bound the eigenvalues as follows.

▶ **Lemma 17.** *Let $G \in \mathcal{G}_k$ have adjacency matrix $A \in \mathbb{F}_2^{k\times k}$. Then*

$$(\lambda_G^{\mathrm{Com}})^2 \le \lambda_G^{\mathrm{Piv}} = 2^{-\mathrm{rank}_{\mathbb{F}_2}(A)}.$$

Proof. We first compute λ_G^{Piv}. Recall that for $S, T \subseteq [k]$, if x_S and x_T are the indicator vectors of S and T, then $x_S^\top A x_T = |\{(u,v) \mid u \in S, v \in T, uv \in E(G)\}|$ over $\mathbb{R}$. For example, $x_S^\top A x_S = 2|E(G) \cap E(K_S)|$. We observe that $|E(G) \cap E(K_{S,T}^\triangle)| \equiv x_S^\top A x_T \pmod 2$. This is because every edge from S to T is counted, with edges in $S \cap T$ counted twice. Using this and Lemma 16,

$$\lambda_G^{\mathrm{Piv}} = \mathbb{E}\left[(-1)^{|E(G \cap K_{S,T}^\triangle)|}\right] = 2^{-2k} \sum_{S,T \subseteq [k]} (-1)^{x_T^\top A x_S} = 2^{-2k} \sum_{u \in \mathbb{F}_2^k} \left(\sum_{v \in \mathbb{F}_2^k} (-1)^{v^\top A u} \right).$$

Consider the inner sum. If $u \in \mathrm{Null}_{\mathbb{F}_2}(A)$, then (over $\mathbb{F}_2$) we have $Au = 0$, so the inner sum is 2^k. Otherwise, Au is a nonzero vector, so exactly half of vectors $v \in \mathbb{F}_2^k$ have $v^\top A u = 0$, and hence the inner sum is 0. Thus,

$$\lambda_G^{\mathrm{Piv}} = 2^{-2k}\Big(2^k |\mathrm{Null}(A)|\Big) = 2^{-k} \cdot 2^{k - \mathrm{rank}(A)} = 2^{-\mathrm{rank}(A)}.$$

Next, we use this to bound λ_G^{Com}. For subsets $S, T \subseteq [k]$, let $R = S \triangle T$. Observe the following two identities:

$$\begin{aligned} |E(G) \cap E(K_S)| + |E(G) \cap E(K_T)| &\equiv |E(G) \cap E(K_{S \triangle T})| + |E(G) \cap E(K_{S,T}^\triangle)| \pmod 2 \\ &\equiv |E(G) \cap E(K_R)| + |E(G) \cap E(K_{S,R}^\triangle)| \pmod 2. \end{aligned}$$

The first can be checked combinatorially, or using linear algebra (over $\mathbb{R}$):

$$\frac{x_S^\top A x_S}{2} + \frac{x_T^\top A x_T}{2} = \frac{(x_S + x_T)^\top A (x_S + x_T)}{2} - x_T^\top A x_S.$$

Evaluating each side mod 2 yields the first identity. For the second, note that by definition $K_{S,T}^\triangle = K_{S,R}^\triangle$, or using algebra $x_T^\top A x_S \equiv (x_T + x_S)^\top A x_S \pmod 2$. As a result, we have the following change of variables (where again we use Lemma 16):

$$\begin{aligned} (\lambda_G^{\mathrm{Com}})^2 = \mathbb{E}\big[(-1)^{|E(G) \cap E(K_S)|}\big]^2 &= 2^{-2k} \sum_{S \subseteq [k]} (-1)^{|E(G) \cap E(K_S)|} \sum_{T \subseteq [k]} (-1)^{|E(G) \cap E(K_T)|} \\ &= 2^{-2k} \sum_{S,T \subseteq [k]} (-1)^{|E(G) \cap E(K_{S \triangle T})|} (-1)^{|E(G) \cap E(K_{S,T}^\triangle)|} \\ &= 2^{-2k} \sum_{R \subseteq [k]} (-1)^{|E(G) \cap E(K_R)|} \sum_{S \subseteq [k]} (-1)^{|E(G \cap K_{S,R}^\triangle)|} \\ &\leq 2^{-2k} \sum_{R \subseteq [k]} \left| \sum_{S \subseteq [k]} (-1)^{x_S^\top A x_R} \right| = \lambda_G^{\mathrm{Piv}}. \end{aligned}$$

In the last line, we used the fact again that the inner sum is either 2^k or 0. Thus, we conclude that $(\lambda_G^{\mathrm{Com}})^2 \leq \lambda_G^{\mathrm{Piv}} = 2^{-\mathrm{rank}(A)}$, as desired. ◀

▶ Remark 18. We note that A is an alternating matrix over $\mathbb{F}_2$, and so the rank of A is even. Thus in particular, if $G = I_k$, $\lambda_G^{\mathrm{Com}} = \lambda_G^{\mathrm{Piv}} = 1$, and for every other $G \in \mathcal{G}_k$, $\lambda_G^{\mathrm{Com}} \leq 1/2$ and $\lambda_G^{\mathrm{Piv}} \leq 1/4$.

We note that μ_{unif} is a multiple of χ_{I_k}, and so the crude bound of $1/2$ is enough to get exponential convergence to the uniform distribution. To get even sharper bounds, we count the number of graphs whose adjacency matrix has a given rank. This is explicitly enumerated by MacWilliams [29], but we only require the following bound.

▶ **Theorem 19** ([29, Theorem 3]). *Let $\mathcal{G}_{k,r}$ denote the set of labeled graphs whose adjacency matrix (a symmetric matrix in $\mathbb{F}_2^{k\times k}$ with 0 diagonal) has rank r. Then if $r \geq 1$ is even,*

$$|\mathcal{G}_{k,r}| \leq 2^{rk-2}$$

and if r is odd, $|\mathcal{G}_{k,r}| = 0$.

We are now ready to show that the two random walks (or any combination of the two) converge exponentially fast to the uniform distribution.

▶ **Lemma 20.** *Let $\mu_0 = \delta_{I_k}$ be the point mass distribution on the independent graph $I_k \in \mathcal{G}_k$. Then for $t = 1, \ldots, \ell$, define either $\mu_t = \mathrm{Com}(\mu_{t-1})$ or $\mu_t = \mathrm{Piv}(\mu_{t-1})$. Suppose that the* Com *transformation is applied m_1 times and the* Piv *transformation is applied m_2 times, and let $m = m_1 + 2m_2$. Then if $m > 2k$, we have*

$$|\mu_\ell(H) - \mu_{\mathrm{unif}}(H)| \leq 2^{2k-\binom{k}{2}-m}$$

for every graph $H \in \mathcal{G}_k$.

Proof. Note that $\{\chi_G \mid G \in \mathcal{G}_k\}$ is an orthogonal basis of $\mathbb{R}^{\mathcal{G}_k}$. Because $\mu_0 = \delta_{I_k}$ has the same inner product with each eigenvector, we have exactly $\mu_0 = 2^{-\binom{k}{2}} \sum_{G\in\mathcal{G}_k} \chi_G$. By induction on t, we compute

$$\mu_\ell = 2^{-\binom{k}{2}} \sum_{G\in\mathcal{G}_k} (\lambda_G^{\mathrm{Com}})^{m_1} (\lambda_G^{\mathrm{Piv}})^{m_2} \chi_G.$$

We note that $2^{-\binom{k}{2}}\chi_{I_k} = \mu_{\mathrm{unif}}$ and $\lambda_{I_k}^{\mathrm{Com}} = \lambda_{I_k}^{\mathrm{Piv}} = 1$, so

$$\mu_\ell - \mu_{\mathrm{unif}} = 2^{-\binom{k}{2}} \sum_{G\neq I_k} (\lambda_G^{\mathrm{Com}})^{m_1} (\lambda_G^{\mathrm{Piv}})^{m_2} \chi_G.$$

We now decompose the sum into the families $\mathcal{G}_{k,2r}$ for all $r \geq 0$, noting that $\mathcal{G}_{k,0} = \{I_k\}$. Using the eigenvalue bounds in Lemma 17 and the counting bounds of Theorem 19, we have the following for any $H \in \mathcal{G}_k$:

$$\begin{aligned}
|\mu_\ell(H) - \mu_{\mathrm{unif}}(H)| &\leq 2^{-\binom{k}{2}} \sum_{G\neq I_k} |\lambda_G^{\mathrm{Com}}|^{m_1} \cdot |\lambda_G^{\mathrm{Piv}}|^{m_2} \cdot |\chi_G(H)| \\
&\leq 2^{-\binom{k}{2}} \sum_{r=1}^{\lfloor k/2 \rfloor} \sum_{G\in\mathcal{G}_{k,2r}} 2^{-rm_1} 2^{-2rm_2} \\
&\leq 2^{-\binom{k}{2}} \sum_{r=1}^{\infty} 2^{2rk-2} 2^{-rm} \\
&\leq 2^{-\binom{k}{2}+2k-m},
\end{aligned}$$

where we use that $m > 2k$ so the sum converges. ◀

We will use Lemma 20 to show that, in a graph $G \sim \mathbb{G}(n, 1/2)$ with $U \subseteq V(G)$, by repeatedly locally complementing/pivoting at distinct vertices outside U, the distribution of $G \circ (x_t)_t[U]$ becomes quickly uncorrelated from the initial state $G[U]$. However, to apply Lemma 20, we need that the local complementation/pivot steps are actually independent; this relies on the initial graph being uniformly random. In fact, we need that the steps of the random walk are independent even when we choose the sequence $(x_t)_t$ based on $G - U$. Towards that end, we prove the following technical lemma.

▶ **Lemma 21.** *Let $W \subseteq \hat{V} \subseteq [n]$ and $U = [n] \setminus \hat{V}$, and fix a graph $\hat{G}$ with vertex set $\hat{V}$. Let $v_1, \dots, v_t \in W$ be a (possibly repeating) sequence of vertices depending on $\hat{G}$. Let G be a random graph with vertex set $[n]$ drawn from $\mathbb{G}(n, 1/2)$ conditioned on $G[\hat{V}] = \hat{G}$. Partition $\binom{[n]}{2}$ into $E_1 = \binom{\hat{V}}{2} \cup [W, U]$ and $E_2 = \binom{U}{2} \cup [\hat{V} \setminus W, U]$. Let $G' = G * (v_i)_{i=1}^t$. Then $E(G') \cap E_2$ is uniformly random and independent of $E(G) \cap E_1$.*

We think of our random walk as first revealing the edges of $G[\hat{V}] = \hat{G}$ to choose the sequence $v_1, \dots, v_t$. As we perform the random walk, we reveal edges between the corresponding vertices $v_1, \dots, v_t$ and U. The set W will correspond to these revealed vertices. Lemma 21 shows that the next step in our random walk is independent of all revealed information.

Proof of Lemma 21. Fix a subset A of E_1. It suffices to show that conditioned the event that $E(G) \cap E_1 = A$, the distribution of $E(G') \cap E_2$ is uniformly random. We proceed by induction on t. If $t = 0$ then $G' = G$, so by definition of $\mathbb{G}(n, 1/2)$, $E(G') \cap E_2$ is uniformly random.

Suppose $t \geq 1$ and let $G'' = G * (v_i)_{i=1}^{t-1}$; by the induction hypothesis, $E(G'') \cap E_2$ is uniformly random conditioned on $E(G) \cap E_1$. Observe that $G' = G'' * v_t$ is obtained by the involutive map $H \mapsto H \triangle K_T$, where $T = N_{G''}(v_t)$ is determined exactly by $E(G) \cap E_1$. This induces an involutive map $E(G'') \cap E_2 \mapsto E(G') \cap E_2$, and therefore preserves the uniform measure. ◀

▶ Remark 22. We note that because the above proposition holds for every graph $\hat{G}$, it also holds if we let $\hat{G}$ be uniformly random, and thus it holds when $G \sim \mathbb{G}(n, 1/2)$ and $v_1, \dots, v_t$ is a random sequence depending on $G[\hat{V}]$.

Observe that Lemma 21 implies that conditioning on $G - U$ and some previous mixing steps $x_1, \dots, x_{t-1} \subseteq W$, as long as x_t is disjoint from W and U, we have that both $G[U]$ and the edges from x_t to U are uniformly random. This implies that each step of our random walk is independent of the previous steps. With this, we conclude the final random walk result.

▶ **Lemma 23.** *Let $G \sim \mathbb{G}(n + k, 1/2)$ with partition $V(G) = \hat{V} \sqcup U$, $|U| = k$. Let $\hat{G}$ be a graph on vertex set $\hat{V}$ and let $(x_i)_{i=1}^{\ell}$ be a sequence of disjoint singletons and pairs from $\hat{V}$ such that $\hat{G} \circ (x_i)_{i=1}^{\ell}$ is well-defined and $|\bigcup_i x_i| > 2k$. Then,*

$$\left| \mathbb{P}\left[G \circ (x_i)_{i=1}^{\ell}[U] = G[U] \;\middle|\; G[\hat{V}] = \hat{G} \right] - 2^{-\binom{k}{2}} \right| \leq 2^{2k - \binom{k}{2} - |\bigcup_i x_i|}.$$

Proof. For times $t = 0, 1, \dots, \ell$, define $G_t \coloneqq G \circ (x_i)_{i=1}^{t}$, and define the probability distribution $\mu_t \in [0,1]^{\mathcal{G}_k}$ as the distribution of $H_t := (G_t \triangle G)[U]$, conditioned on $G[\hat{V}] = \hat{G}$. For example, μ_0 is the point mass distribution δ_{I_k}.

At time t, H_{t-1} is a function of $G[\hat{V}] = \hat{G}$ and $N_G(w)$ for $w \in \bigcup_{i=1}^{t-1} x_i$. If $x_t = \{v\}$, then $H_t = H_{t-1} \triangle K_S$ where $S = N_{G_{t-1}}(v) \cap U$. Similarly, if $x_t = \{v, v'\}$, then $H_t = H_{t-1} \triangle K^{\triangle}_{S,T}$, where $S = N_{G_{t-1}}(v) \cap U$ and $T = N_{G_{t-1}}(v') \cap U$. By Lemma 21 (applied with $W = \bigcup_{i<t} x_i$), we have that S (resp. T) is a uniformly random subset of U, independent of $G[\hat{V}]$ and $N_G(u)$ for $u \in \bigcup_{i<t} x_i$ and thus S (resp. T) is independent of H_{t-1}. Then by definition, if $|x_t| = 1$ then we have exactly $\mu_t = \mathrm{Com}(\mu_{t-1})$, and if $|x_t| = 2$ then $\mu_t = \mathrm{Piv}(\mu_{t-1})$. We then apply Lemma 20 (noting that the x_i are disjoint so $m = |\bigcup_i x_i|$) to conclude

$$\left| \mathbb{P}\left[(G_\ell \triangle G)[U] = I_k \;\middle|\; G[\hat{V}] = \hat{G} \right] - 2^{-\binom{k}{2}} \right| = |\mu_\ell(I_k) - \mu_{\mathrm{unif}}(I_k)| \;\leq\; 2^{2k - \binom{k}{2} - |\bigcup_i x_i|},$$

as desired. ◀

3.2 Proof of Theorem 3

We now use the reordering Lemma 12 to show that when J, J' are sufficiently different, the graphs G_J and $G_{J'}$ are sufficiently uncorrelated by applying the random walk described in Section 3.1.

▶ **Lemma 24.** *Let $G \sim \mathbb{G}(n+k, 1/2)$, $U \subseteq V(G)$ with $|U| = k$, and let $J, J' \subseteq [n]$ with $|J \triangle J'| > 2k$. Defining $G_J, G_{J'}$ as in Equation (1), we have*

$$\left|\mathbb{P}[G_J[U] = G_{J'}[U]] - 2^{-\binom{k}{2}}\right| \leq 2^{2k-\binom{k}{2}-|J\triangle J'|}$$

Proof. Let $J = \{i_1 < \ldots < i_\ell\}$ and $J' = \{j_1 < \ldots < j_{\ell'}\}$. Observe that

$$G_{J'} = G_J * v_{i_\ell} * \cdots * v_{i_2} * v_{i_1} * v_{j_1} * v_{j_2} * \cdots * v_{j_{\ell'}}. \tag{2}$$

Fix $\hat{G} \in \mathcal{G}_n$, and condition on the event $G[\hat{V}] = \hat{G}$, which depends only on the edges inside $\hat{V}$. By the reordering Lemma 12, there is a sequence $(x_i)_{i=1}^r$ (depending on $\hat{G}$) with the three properties from that lemma such that $G_{J'}[U] = G_J \circ (x_i)_{i=1}^r[U]$. By property (III), we note that $|\bigcup_i x_i| \geq |J \triangle J'|$, since a vertex v_i appears exactly once in Equation (2) if $i \in J \triangle J'$. Let $\hat{G}_J = \hat{G} * (v_{i_t})_{t=1}^\ell$, so $G[\hat{V}] = \hat{G}$ if and only if $G_J[\hat{V}] = \hat{G}_J$. Then by Lemma 23,

$$\begin{aligned}\left|\mathbb{P}\left[G_{J'}[U] = G_J[U] \;\middle|\; G[\hat{V}] = \hat{G}\right] - 2^{-\binom{k}{2}}\right| &= \left|\mathbb{P}\left[G_J \circ (x_i)_{i=1}^r[U] = G_J[U] \;\middle|\; G_J[\hat{V}] = \hat{G}_J\right] - 2^{-\binom{k}{2}}\right| \\ &\leq 2^{2k-\binom{k}{2}-|\bigcup_{i\leq r} x_i|} \\ &\leq 2^{2k-\binom{k}{2}-|J\triangle J'|},\end{aligned}$$

as desired. We apply the law of total probability over all $\hat{G} \in \mathcal{G}_n$ to obtain the unconditioned bound. ◀

Before we complete the second moment method calculation, we need one more fact about the independence implicit in our construction.

▶ **Proposition 25.** *Let $G \sim \mathbb{G}(n+k, 1/2)$, $U \subseteq V(G)$ with $|U| = k$, and let $J, J' \subseteq [n]$. Defining $G_J, G_{J'}$ as in Equation (1), the graphs $G_J[U]$ and $(G_J \triangle G_{J'})[U]$ are independent.*

Proof. We use a different but equivalent sampling procedure to obtain G_J and $G_{J'}$. First, sample $G_J \sim \mathbb{G}(n, 1/2)$ uniformly at random, which is the correct distribution by Proposition 13. In particular, the sets $E(G_J[U])$ and $E(G_J) \setminus E(G_J[U])$ are independent. Next, compute $G_{J'}$ directly using Equation (2). Observe that the set $E(G_J \triangle G_{J'})$ is a function only of $E(G_J) \setminus E(G_J[U])$ since this procedure for obtaining $G_{J'}$ depends on the edges incident to the v_i's. Thus the graphs $G_J[U]$ and $(G_J \triangle G_{J'})[U]$ are independent, as desired. ◀

We now perform the second moment calculation.

▶ **Lemma 26.** *Let $G \sim \mathbb{G}(n+k, 1/2)$. Fix a vertex set $U \subseteq V(G)$ with $|U| = k$ and a graph H on U. Then H is a vertex-minor of G with probability at least $1 - 2^{2k}(3/4)^n - 2^{-n+\binom{k}{2}+2k\log_2 n}$.*

Proof. For all $J \subseteq [n]$, construct G_J as in (1). Let X_J be the indicator that $G_J[U] = H$, and let $X = \sum_{J\subseteq[n]} X_J$. Recall that $\mathbb{E}[X] = 2^{n-\binom{k}{2}}$ by linearity of expectation and Proposition 13. We compute

$$\begin{aligned}\mathrm{Var}[X] &= \sum_{J,J'\subseteq[n]} \mathbb{E}[X_J X_{J'}] - \mathbb{E}[X_J]\mathbb{E}[X_{J'}] \\ &= \sum_{J,J'\subseteq[n]} \mathbb{P}[G_J[U] = H \wedge G_J[U] = G_{J'}[U]] - \mathbb{P}[G_J[U] = H]\mathbb{P}[G_{J'}[U] = H] \\ &= \sum_{J,J'\subseteq[n]} \mathbb{P}[G_J[U] = H]\mathbb{P}[G_J[U] = G_{J'}[U]] - \mathbb{P}[G_J[U] = H]\mathbb{P}[G_{J'}[U] = H],\end{aligned}$$

where we use the independence from Proposition 25. Recall that $G_J[U], G_{J'}[U]$ are both uniformly random, so by Lemma 24 we have

$$\begin{aligned}\mathrm{Var}[X] &= \sum_{J\subseteq[n]} 2^{-\binom{k}{2}} \sum_{J'\subseteq[n]} \left(\mathbb{P}[G_J[U] = G_{J'}[U]] - 2^{-\binom{k}{2}}\right) \\ &= \sum_{J\subseteq[n]} 2^{-\binom{k}{2}} \sum_{J'\triangle J\subseteq[n]} \left(\mathbb{P}[G_J[U] = G_{J'}[U]] - 2^{-\binom{k}{2}}\right) \\ &\leq \sum_{J\subseteq[n]} 2^{-\binom{k}{2}} \left(\sum_{\substack{J'\triangle J\subseteq[n] \\ |J'\triangle J|>2k}} 2^{2k-\binom{k}{2}-|J'\triangle J|} + \sum_{\substack{J'\triangle J\subseteq[n] \\ |J'\triangle J|\leq 2k}} 1 \right) \\ &= \sum_{J\subseteq[n]} 2^{-\binom{k}{2}} \left(\sum_{i=2k+1}^{n} \binom{n}{i} 2^{2k-\binom{k}{2}-i} + \sum_{i=0}^{2k} \binom{n}{i} \right) \\ &\leq \sum_{J\subseteq[n]} 2^{-\binom{k}{2}} \left(2^{2k-\binom{k}{2}} \sum_{i=0}^{n} \binom{n}{i} 2^{-i} + n^{2k} \right) \\ &\leq \sum_{J\subseteq[n]} 2^{-\binom{k}{2}} \left(2^{2k-\binom{k}{2}} (3/2)^n + n^{2k} \right) \\ &= 2^{n-\binom{k}{2}} \left(2^{2k-\binom{k}{2}} (3/2)^n + n^{2k} \right).\end{aligned}$$

(To obtain the bound $\sum_{i=0}^{2k} \binom{n}{i} \leq n^{2k}$, consider for example the surjection $f : [n]^{2k} \to \binom{n}{\leq 2k}$ given by $f(x_1, \ldots, x_{2k}) = \{x_1, \ldots, x_{2k}\}$, except $f(2k, 2k-1, \ldots, 1) = \emptyset$.) In conclusion, by Chebyshev's inequality (Lemma 14) we have

$$\mathbb{P}[X = 0] \leq \frac{\mathrm{Var}[X]}{\mathbb{E}[X]^2} \leq \frac{2^{n-\binom{k}{2}} \left(2^{2k-\binom{k}{2}} (3/2)^n + n^{2k} \right)}{2^{2n-2\binom{k}{2}}} = 2^{2k}(3/4)^n + 2^{-n+\binom{k}{2}} n^{2k}$$

as desired. ◀

We are now ready to prove our main result via the union bound over all choices of $U \subseteq V(G)$ and graphs H with vertex set U. We first restate the theorem for convenience.

▶ **Theorem 3.** *For all $c > 0$, if $G \sim \mathbb{G}(n, 1/2)$ with $n \geq (1+c)\frac{1}{2\log_2(4/3)}k^2$, then G is k-vertex-minor universal with probability at least $1 - 2^{-(1+o(1))ck^2/2}$.*

Proof. Let $G \sim \mathbb{G}(n, 1/2)$, where $n = (1+c)\frac{1}{2\log_2(4/3)}k^2$. Here we allow $c = c(k)$ to be increasing with k, but it is lower bounded by some positive constant. By Lemma 26, the probability that a given graph H on k given vertices of $V(G)$ is not a vertex-minor of G is at most $2^{2k}(3/4)^{n-k} + 2^{-n+k+\binom{k}{2}}(n-k)^{2k} = 2^{2k}(3/4)^{(1+o(1))n}$.

By the union bound, we have

$$\begin{aligned}\mathbb{P}[G \text{ is not } k\text{-vertex-minor universal}] &\leq \binom{n}{k} 2^{\binom{k}{2}} \cdot \left(2^{2k}(3/4)^{(1+o(1))n}\right) \qquad (3)\\ &\leq \exp_2\left(k\log_2 n + \frac{k^2}{2} + 2k - (1+o(1))\log_2(4/3)n\right)\\ &= \exp_2\left(O(k\log(ck)) + \frac{k^2}{2} - (1+o(1))\frac{1+c}{2}k^2\right)\\ &= \exp_2\left(O(k\log(ck)) - (1+o(1))\,ck^2/2\right).\end{aligned}$$

Thus, whenever $n = (1+c)\frac{1}{2\log_2(4/3)}k^2$, we have $\mathbb{P}[G \text{ is } k\text{-vm-universal}] \geq 1 - 2^{-(1+o(1))ck^2/2}$, as desired. ◀

3.3 Proof of reordering Lemma 12

First, we restate the vertex-minor reordering lemma:

▶ **Lemma 12.** *Fix a set $\hat{V}$. Let $v_1, \ldots, v_\ell \in \hat{V}$ be a (possibly repeating) sequence and let $\hat{G}$ be a graph with $V(\hat{G}) = \hat{V}$. Then there is a sequence $x_1, \ldots, x_r$ of singletons and pairs from $\hat{V}$ such that for any graph G with $\hat{V} \subseteq V(G)$ and $G[\hat{V}] = \hat{G}$, we have*

- **(I)** *$G * v_1 * v_2 \cdots * v_\ell - \hat{V} = G \circ x_1 \cdots \circ x_r - \hat{V}$.*
- **(II)** *The sets $x_1, \ldots, x_r$ are pairwise disjoint.*
- **(III)** *If $v \in \hat{V}$ appears exactly once in $(v_i)_{i=1}^{\ell}$, then $v \in x_j$ for some $j \in [r]$.*

Moreover, any choice of $(x_i)_i$ satisfying (I) and (II) for all G also satisfies (III).

We note that if $\hat{V}$ is an independent set in G, the complementations $*v_i$ are commutative and self-inverse, so we can choose $(x_i)_i$ to be any ordering of those vertices v_i which appear in the sequence an odd number of times. If not, then the complementations interact in some fixed way determined only by $G[\hat{V}]$. The proof idea is to pull out one *special* vertex $v \in \hat{V}$ at a time, and use Theorem 11 to force that v appears at most once in the resulting sequence $(x_i)_i$. We do this via the following inductive lemma in which the ordering may depend on the host graph G. We will then show that the ordering does in fact work for all graphs G with $G[\hat{V}] = \hat{G}$.

▶ **Lemma 27.** *For any graph G with $V(G) = \hat{V} \sqcup U$ and $v_1, \ldots, v_\ell \in \hat{V}$, and for any special vertex $v \in \hat{V}$, there exists a sequence $(y_i)_{i=1}^{r}$ satisfying (I) and (II) for G such that one of the following holds:*

1. *we have $v \notin \bigcup_i y_i$,*
2. *we have $v \in y_1$, or*
3. *we have $y_2 = \{v\}$ and $y_1 = \{w\}$ for some $w \in \hat{V}$ with $vw \in E(G)$.*

Proof. Let $H = G * v_1 * \cdots * v_\ell[U]$, so H is a vertex-minor of G. We proceed by induction on $n := |V(G)|$; the base case $n = 1$ is trivial. Let $v \in \hat{V}$ be arbitrary.

By Theorem 11, H is a vertex-minor of $G - v$, $G * v - v$, or $G \times vw - v$ for some w adjacent to v. In the first case, we apply the induction hypothesis to $G - v$, and clearly the resulting sequence $(y_i)_i$ satisfies item 1. In the second case, we apply the induction hypothesis to $G * v - v$ to obtain the sequence $(y_i)_{i\geq 2}$. Then clearly setting $y_1 = \{v\}$ satisfies item 2.

In the final case, we apply the induction hypothesis to $G \times vw - v$ for some w adjacent to v such that w is the special vertex to obtain the sequence $(y_i)_{i\geq 2}$. We then split into cases based on the location of w. If $w \notin \bigcup_{i\geq 2} y_i$, then setting $y_1 = \{v, w\}$ gives the desired sequence

$(y_i)_{i\geq 1}$ satisfying item 2. If $y_2 = \{w\}$, then note $G\times vw*w = G*w*v$, and so replacing y_2 with $\{v\}$ and setting $y_1 = \{w\}$ gives the desired sequence $(y_i)_{i\geq 1}$ satisfying item 3. If $y_2 = \{w, u\}$, then by Lemma 10, $G \times vw \times wu = G \times vu$, and so replacing y_2 with $\{v, u\}$ yields the desired sequence $(y_i)_{i\geq 2}$ satisfying item 2. Finally, if $y_2 = \{u\}, y_3 = \{w\}$ with $uw \in E(G \times vw)$, then we use the fact that $G \times vw * u * w = G \times vw \times uw * u = G \times vu * u = G * u * v$, and so by replacing y_2, y_3 with $\{u\}, \{v\}$ respectively, we get the desired sequence $(y_i)_{i\geq 2}$ satisfying item 3. ◀

Finally, to construct a sequence $(x_i)_i$ which satisfies Lemma 12 for all graphs G with $G[\hat{V}]$, we apply 27 to a special gadget which encodes all possible structures that can appear in G.

Proof of Lemma 12. Let $\hat{V}$ be a set, and fix a sequence $v_1, \ldots, v_\ell \in \hat{V}$ and a graph $\hat{G}$ on vertex set $\hat{V}$. To show that we can pick $(x_i)_i$ to depend only on $\hat{G}$, we apply Lemma 27 to a universal "gadget" graph defined as follows. Let $\mathbf{G}$ have vertex set $\hat{V} \cup [3 \cdot 2^{3|\hat{V}|}]$ by setting $\mathbf{G}[\hat{V}] = \hat{G}$, placing $2^{3|\hat{V}|}$ disjoint copies of the 3 vertex path P_3, and adding edges from $\hat{V}$ to the copies of P_3 such that all of the $2^{3|\hat{V}|}$ possible sets of edges occur between $\hat{V}$ and a copy of P_3. This has the property that for any graph G on vertex set $\hat{V} \cup \{u_1, u_2\}$ with $G[\hat{V}] = \hat{G}$, there exist j_1, j_2 such that $\mathbf{G}[\hat{V} \cup \{j_1, j_2\}]$ is isomorphic to G with the isomorphism $\hat{V} \mapsto \hat{V}$ and $u_i \mapsto j_i$. Such an isomorphism can be found by taking the copy of P_3 with the correct adjacency to $\hat{V}$ and letting j_1, j_2 be either adjacent or nonadjacent vertices in the P_3.

▷ Claim 12.1. For all $k \geq 2$, if the sequence $(x_i)_{i=1}^r$ satisfies properties (I) and (II) for the gadget $\mathbf{G}$, then it satisfies property (III).

Proof. Suppose $v \in \hat{V}$ appears exactly once in $\{v_i\}_i$ in position i^*, and suppose $(x_i)_i$ does not contain v. Expanding out $\circ(x_i)_i = *(w_i)_{i=1}^m$ as a sequence of local complementations for $w_i \in \hat{V} \setminus \{v\}$, for $\mathbf{G}' = \mathbf{G} * v_1 * \cdots * v_{i^*-1}$, we have

$$\mathbf{G}' * v = \mathbf{G}' * v_{i^*-1} * \cdots * v_1 * w_1 * \cdots * w_m * v_\ell * \cdots * v_{i^*+1}. \tag{4}$$

We show this is impossible. Let RHS denote the right-hand side of (4). Choose vertices $j_1, j_2 > n$ such that $j_1 j_2 \notin E(\mathbf{G})$ and $N_{\mathbf{G}}(j_1) \cap \hat{V} = N_{\mathbf{G}}(j_2) \cap \hat{V} = \{v\}$; these vertices exist by definition of $\mathbf{G}$. Then we note $j_1 j_2 \notin E(\mathbf{G}')$ and $N_{\mathbf{G}'}(j_1) = N_{\mathbf{G}}(j_1)$, $N_{\mathbf{G}'}(j_2) = N_{\mathbf{G}}(j_2)$. Then $j_1 j_2 \in E(\mathbf{G}' * v)$ but $j_1 j_2 \notin E(\text{RHS})$, because no local complementation step can add an edge incident to j_1, j_2. This contradicts (4), so $(x_i)_i$ contains v. ◁

We now show that if a sequence satisfies Lemma 27 for $\mathbf{G}$, then it satisfies Lemma 12. Let $(x_i)_{i=1}^r$ be the sequence given by Lemma 27 for the gadget $\mathbf{G}$. Let G be an arbitrary graph with $\hat{V} \subseteq V(G)$ and $G[\hat{V}] = \hat{G}$. We need to show that the sequence satisfies condition (I) for G. For this it suffices to show that for all $u_1, u_2 \in V(G) \setminus \hat{V}$, we have $u_1 u_2 \in E(G * (v_i)_{i=1}^\ell)$ if and only if $u_1 u_2 \in E(G \circ (x_i)_{i=1}^j)$. Indeed, letting $G' = G[\hat{V} \cup \{u_1, u_2\}]$, there exist $j_1, j_2 \in V(\mathbf{G})$ such that $\mathbf{G}[\hat{V} \cup \{j_1, j_2\}]$ is isomorphic to G with isomorphism $\hat{V} \mapsto \hat{V}$ and $u_i \mapsto j_i$. In particular, this means $G' * (v_i)_i - \hat{V} = G' \circ (x_i)_i - \hat{V}$, so we conclude that $u_1 u_2 \in E(G' * (v_i)_i)$ if and only if $u_1 u_2 \in E(G' * (v_i)_i)$, as desired. Thus we conclude $G * (v_i)_i - \hat{V} = G \circ (x_i)_i - \hat{V}$, so all three properties hold. ◀

4 Vertex-minor Ramsey and concluding remarks

We now prove our bounds on the vertex-minor Ramsey problem stated in the introduction.

▶ **Theorem 8.** *For all $k \geq 0$,*

$$(1+o(1))\frac{1}{2\log_2 3}k^2 \leq R_{\mathrm{vm}}(k) \leq 2^k - 1.$$

Proof. We first prove the upper bound. When $k = 0, 1$ this is satisfied with equality. If $k \geq 2$, let $N = R_{\mathrm{vm}}(k) - 1$, so there is a graph G on N vertices with $U \subseteq V(G)$ an independent set in G such that $|U| \leq k-1$ and G does not contain a larger independent set as a vertex-minor. For all $S \subseteq U$, let $V_S = \{v \in V(G) \setminus U : N(v) \cap U = S\}$. Then we have the disjoint union $V(G) = U \cup \bigcup_{S \subseteq U} V_S$. We now claim that $|V_S| \leq \min\{2, |S|\}$ for all $S \subseteq U$. Assuming this, we conclude

$$R_{\mathrm{vm}}(k) - 1 = N \leq (k-1) + 1 \cdot \binom{k-1}{1} + 2 \cdot (2^{k-1} - k) = 2^k - 2.$$

Note that if $x \in V_\emptyset$, then $U \cup \{x\}$ is an independent set, contradicting the maximality of U. Suppose towards a contradiction that $x, y \in V_{\{u\}}$ for some $u \in U$. By locally complementing at u, we may assume that $xy \notin E(G)$. Then $U \cup \{x, y\} \setminus \{u\}$ is an independent set, contradicting the maximality of U.

Suppose towards a contradiction that $|S| \geq 2$ and $x, y, z \in V_S$. Let $u \in S$. By locally complementing at u, we may assume that $G[\{x, y, z\}]$ contains at most 1 edge. If $\{x, y, z\}$ is an independent set in G, then $U \cup \{y, z\} \setminus \{u\}$ is an independent set in $G \times ux$. Otherwise without loss of generality $yz \in E(G)$. Then $U \cup \{y, z\} \setminus \{u\}$ is an independent set in $G * x * u$. In every case we obtain an independent set of size larger than U as a vertex-minor, a contradiction.

For the lower bound, let $G \sim \mathbb{G}(n, 1/2)$. Note that G contains I_k as a vertex-minor if and only if G is locally equivalent (l.e.) to some graph H with $\alpha(H) \geq k$. A result of Bahramgiri and Beigi [2] states that for any graph H on n vertices, there are at most 3^n graphs locally equivalent to H. Then, by a union bound,

$$\begin{aligned}\mathbb{P}[G \text{ l.e. } H,\ \alpha(H) \geq k] &\leq 3^n \mathbb{P}[\alpha(G) \geq k] \\ &\leq 3^n \binom{n}{k} 2^{-\binom{k}{2}} \leq \exp_2\left(n\log_2 3 + k\log_2 n - k^2/2 + k/2\right) < 1,\end{aligned}$$

as long as $n \leq (1+o(1))\frac{1}{2\log_2 3}k^2$. ◀

We note that our upper bound is the correct answer for $k \leq 3$. (For $k = 3$, it is not hard to verify that the wheel on 6 vertices does not contain an independent of size 3 as a vertex-minor.) However, since the lower bound in Theorem 8 comes from the random graph, it is natural to believe that the polynomial lower bound is closer to the truth for sufficiently large k than our modest upper bound. This belief leads us to Conjecture 9, restated below.

▶ **Conjecture 9.** $R_{\mathrm{vm}}(k) \leq \mathsf{poly}(k)$.

By Theorem 3, the random graph $\mathbb{G}(n, 1/2)$ cannot be a counterexample, and indeed it is possible that $R_{\mathrm{vm}}(k) = O(k^2)$. We note that proper vertex-minor closed classes have the Erdős-Hajnal property [10]. This implies that if we restrict our graphs to any proper

vertex-minor closed class, they contain an independent set of size $\Omega(n^c)$ as a vertex-minor for some $c > 0$ possibly depending on the class. We also make the connection between pivot-minor universality and its corresponding Ramsey-type question.

▶ **Definition 28.** *Let $R_{\text{piv}}(k)$ be the minimum n such that any n-vertex graph contains an independent set or clique of size k as a pivot-minor.*

Note that unlike the definition of $R_{\text{vm}}(k)$, we need to allow either a clique or an independent set in this definition since K_n does not contain any non-trivial independent set as a pivot-minor.

We note that trivially $R_{\text{vm}}(k-1) \leq R_{\text{piv}}(k) \leq R(k)$, the diagonal Ramsey number. Theorem 5 supports the following conjecture, which would imply Conjecture 9.

▶ **Conjecture 29.** $R_{\text{piv}}(k) \leq \mathsf{poly}(k)$.

We note that proper pivot-minor closed classes also have the Erdős-Hajnal property [19], so this conjecture holds if we restrict to any proper pivot-minor closed class. We also improve the trivial upper bound of $R(k)$ for $R_{\text{piv}}(k)$.

▶ **Theorem 30.** *For all $k \geq 0$, $R_{\text{piv}}(k) \leq (k-1)(2^k - 1) + 1$.*

Proof. Let G be a graph with no independent set or clique of size k as a pivot-minor on $R_{\text{piv}}(k) - 1$ vertices. By possibly replacing G with a pivot-equivalent graph, we let U be an independent set in G such that G does not contain a larger independent set as a pivot-minor. For all $S \subseteq U$, let $V_S = \{v \in V(G) \setminus U : N(v) \cap U = S\}$. Clearly $V_\emptyset$ is empty by the maximality of U. For S containing a vertex u, we claim that V_S can not contain an independent set of size 3. Indeed otherwise if $\{x, y, z\} \in V_S$ is independent, then $U \cup \{y, x\} \setminus \{u\}$ is an independent set in $G \times ux$. We also note that V_S can not contain an induced path on 3 vertices. Indeed if x, y, z are the vertices of an induced path in V_S in that order, then $U \cup \{y, x\} \setminus \{u\}$ is an independent set in $G \times ux$. Clearly V_S can not contain a clique of size k by definition of G. It follows that $G[V_S]$ is a disjoint union of cliques and thus $|V_S| \leq 2(k-1)$. Thus

$$R_{\text{piv}}(k) - 1 \leq (k-1) + (2^{k-1} - 1)(2k-2) = (k-1)(2^k - 1). \qquad \blacktriangleleft$$

We make some final conjectures about vertex-minor universality. Theorem 3 and our proof of the lower bound in Theorem 8 together show that for any $\varepsilon > 0$, the random graph $G \sim \mathbb{G}(n, 1/2)$ is with high probability $(\alpha - \varepsilon)\sqrt{n}$-vertex-minor universal but not $(\beta + \varepsilon)\sqrt{n}$-vertex-minor universal, where $\alpha = \sqrt{2\log_2(4/3)} \approx 0.911$ and $\beta = \sqrt{2\log_2(3)n} \approx 1.78$. It is natural to conjecture that there is a sharp threshold for universality, as follows.

▶ **Conjecture 31.** *There exists $C \in [\sqrt{2\log_2(4/3)}, \sqrt{2\log_2(3)}]$ such that for any $\varepsilon > 0$, with high probability $G \sim \mathbb{G}(n, 1/2)$ is $(C - \varepsilon)\sqrt{n}$-vertex-minor universal and not $(C + \varepsilon)\sqrt{n}$-vertex-minor universal.*

It is also natural to ask for what values of p the random graph $\mathbb{G}(n, p)$ is k-vertex-minor universal for n polynomial in k. The main difficulty in extending our approach to $p \neq 1/2$ is that the random walk defined in Section 3.1 now has steps which are correlated to each other: after flipping an edge which was not yet revealed, its probability of existing is now $1 - p$

instead of p. If one could get around this issue, the resulting random walk would then have the second largest eigenvalue bounded in absolute value by the max of $1-2p^2, 1-2(1-p)^2$, and $1-2p(1-p)$. Assuming without loss of generality that $p \leq 1/2$, this means it would suffice to run the random walk for $O(k^2/p^2)$ steps. Thus we conjecture the following.

▶ **Conjecture 32.** *If $p = \omega(1/\sqrt{n})$ is at most $1/2$ and $G \sim \mathbb{G}(n,p)$ or $G \sim \mathbb{G}(n,1-p)$, then G is k-vertex-minor universal with high probability for some $k = \Omega(p\sqrt{n})$.*

If true, this would qualitatively generalize Theorem 3, although unlike Theorem 3 it is not necessarily tight up to a constant for all p. Conjecture 32 would also imply that $\mathbb{G}(n,p)$ is not a counterexample to Conjecture 9 for any p. This follows for instance by noting that if p is outside the range $[n^{-1/4}, 1-n^{-1/4}]$, then $\mathbb{G}(n,p)$ already contains a clique or independent set of size $\Omega(n^{1/4}\log n)$.

After the appearance of an earlier draft of this paper, Chao and Xu [9] verified Conjecture 32 up to a logarithmic factor.

References

1 Ruben Ascoli, Bryce Frederickson, Sarah Frederickson, Caleb McFarland, and Logan Post. Almost all graphs are vertex-minor universal, 2026. Preprint available at arXiv:2602.09049 [quant-ph]. arXiv:2602.09049.

2 M. Bahramgiri and S. Beigi. Enumerating the classes of local equivalency in graphs, 2007. Preprint available at arXiv:0702267 [math.CO]. arXiv:0702267.

3 A. Bouchet. Graphic presentations of isotropic systems. *J. Combin. Theory Ser. B*, 45:58–76, 1988. doi:10.1016/0095-8956(88)90055-X.

4 S. Bravyi, Y. Sharma, M. Szegedy, and R. de Wolf. Generating k EPR-pairs from an n-party resource state. *Quantum*, 8:1348, 2024. doi:10.22331/Q-2024-05-14-1348.

5 H. Buffière, E. Kim, and P. Ossona de Mendez. Shallow vertex minors, stability, and dependence. *Innov. Graph Theory*, 1:87–112, 2024. doi:10.5802/IGT.5.

6 R. Campbell, J. Pascal Gollin, M. Hatzel, O. Kwon, R. McCarty, S. Oum, and S. Wiederrecht. The Erdős-Pósa property for circle graphs as vertex-minors, 2025. Preprint available at arXiv:2506.03973 [math.CO]. arXiv:2506.03973.

7 M. Campos, S. Griffiths, R. Morris, and J. Sahasrabudhe. An exponential improvement for diagonal Ramsey, 2023. Preprint available at arXiv:2303.09521 [math.CO]. arXiv:2303.09521.

8 M. Cautrès, N. Claudet, M. Mhalla, S. Perdrix, V. Savin, and S. Thomassé. Vertex-minor universal graphs for generating entangled quantum subsystems. In *51st Int. Colloquium on Automata, Languages, and Programming (ICALP 2024)*, 2024.

9 T. W. Chao and Z. Xu. Vertex-minor universality of a random graph, 2026. Preprint available at arXiv:2603.13600 [math.CO]. arXiv:2603.13600.

10 M. Chudnovsky and S. Oum. Vertex-minors and the Erdős–Hajnal conjecture. *Discrete Math.*, 341:3498–3499, 2018. doi:10.1016/J.DISC.2018.09.007.

11 N. Claudet. Personal communication, 2025. Personal communication.

12 N. Claudet, M. Mhalla, and S. Perdrix. Small k-pairable states, 2023. Preprint available at arXiv:2309.09956 [quant-ph]. arXiv:2309.09956.

13 N. Claudet and S. Perdrix. Deciding local unitary equivalence of graph states in quasi-polynomial time. In *52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025)*, 2025. doi:10.4230/LIPIcs.ICALP.2025.59.

14 N. Claudet and S. Perdrix. Local equivalence of stabilizer states: a graphical characterisation. In *42nd International Symposium on Theoretical Aspects of Computer Science (STACS 2025)*, 2025. doi:10.4230/LIPIcs.STACS.2025.27.

15 B. Courcelle and S. Oum. Vertex-minors, monadic second-order logic, and a conjecture by Seese. *J. Combin. Theory Ser. B*, 97:91–126, 2007. doi:10.1016/J.JCTB.2006.04.003.

16 A. Dahlberg, J. Helsen, and S. Wehner. How to transform graph states using single-qubit operations: computational complexity and algorithms. *Quantum Sci. Technol.*, 5:045016, 2020.

17 A. Dahlberg, J. Helsen, and S. Wehner. The complexity of the vertex-minor problem. *Inf. Process. Lett.*, 175:106222, 2022. doi:10.1016/J.IPL.2021.106222.

18 A. Dahlberg and S. Wehner. Transforming graph states using single-qubit operations. *Phil. Trans. R. Soc. A*, 376:20170325, 2018.

19 J. Davies. Pivot-minors and the Erdős-Hajnal conjecture. *J. Combin. Theory Ser. B*, 173:257–278, 2025. doi:10.1016/J.JCTB.2025.04.004.

20 P. Diaconis. Random walks on groups: characters and geometry. In *London Math. Soc. Lecture Note Ser.*, pages 120–142. Cambridge Univ. Press, Cambridge, 2003.

21 D. G. Fon-Der-Flaass. On local complementations of graphs. In *Colloquia Mathematica Societatis János Bolyai*, volume 52, pages 257–266. North-Holland, Amsterdam, 1988.

22 J. Geelen, O. Kwon, R. McCarty, and P. Wollan. The grid theorem for vertex-minors. *J. Combin. Theory Ser. B*, 158:93–116, 2023. doi:10.1016/J.JCTB.2020.08.004.

23 S. Ghosh, D. Hangleiter, and J. Helsen. Random regular graph states are complex at almost any depth. *PRX Quantum*, 6:040344, 2025.

24 D. Gross, S. T. Flammia, and J. Eisert. Most quantum states are too entangled to be useful as computational resources. *Phys. Rev. Lett.*, 102:190501, 2009.

25 P. Gupta, N. Ndiaye, S. Norin, and L. Wei. Optimizing the CGMS upper bound on Ramsey numbers, 2024. Preprint available at arXiv:2407.19026 [math.CO]. arXiv:2407.19026.

26 M. Hein, W. Dür, J. Eisert, R. Raussendorf, M. Van den Nest, and H. Briegel. Entanglement in graph states and its applications. In *Proceedings of the International School of Physics "Enrico Fermi," Vol. 162: Quantum Computers, Algorithms and Chaos*, pages 115–218. IOS Press, 2006.

27 T. Kaldenbach and M. Heller. Mapping quantum circuits to shallow-depth measurement patterns based on graph states. *Quantum Sci. Technol.*, 10:015010, 2025.

28 D. Kim and S. Oum. Vertex-minors of graphs: a survey. *Discrete Applied Mathematics*, 351:54–73, 2024. doi:10.1016/J.DAM.2024.03.011.

29 J. MacWilliams. Orthogonal matrices over finite fields. *Amer. Math. Monthly*, 76:152–164, 1969.

30 S. Oum. Rank-width and vertex-minors. *J. Combin. Theory Ser. B*, 95:79–100, 2005. doi:10.1016/J.JCTB.2005.03.003.

31 M. Van den Nest, J. Dehaene, and B. De Moor. Graphical description of the action of local Clifford transformations on graph states. *Phys. Rev. A*, 69:022316, 2004.

32 J. Wu, M. Rossi, H. Kampermann, S. Severini, L. C. Kwek, C. Macchiavello, and D. Bruß. Randomized graph states and their entanglement properties. *Phys. Rev. A*, 89:052335, 2014.

Improved Error Reduction for Weighted PRGs

Ben Chen
School of Computer Science and AI, Tel-Aviv University, Israel

Gil Cohen
School of Computer Science and AI, Tel-Aviv University, Israel

Dean Doron
Faculty of Computer and Information Science, Ben-Gurion University of the Negev, Beer Sheva, Israel

Yuval Khaskelberg
School of Computer Science and AI, Tel-Aviv University, Israel

Amnon Ta-Shma
School of Computer Science and AI, Tel-Aviv University, Israel

Abstract

We devise an error-reduction procedure that transforms a PRG for length-n, width-w read-once branching programs with error $1/\mathrm{poly}(n)$ and seed length s_0, over any alphabet, into a weighted PRG with seed length

$$s_0 + O\left(\log\frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right)\cdot\log w\right).$$

Using this reduction, we improve upon the state-of-the-art weighted PRG constructions of Hoza (RANDOM 2021) and Cheng and Wu (SODA 2026), achieving optimal dependence on the program's arity while matching the best known bounds in all other parameters.

Our motivation for obtaining optimal dependence on the arity stems from a result of Cheng and Hoza (CCC 2020, ToC 2022), who showed that a PRG with optimal arity and error dependence yields a PRG with seed length $O(\log^{3/2} n)$ (for, say, constant width), thereby breaking the long-standing log-squared barrier.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Space-bounded computation, pseudorandom generators

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.39

Category RANDOM

Related Version *Full Version*: `https://eccc.weizmann.ac.il/report/2026/064/` [7]

Funding *Ben Chen*: Funded by the Israel Science Foundation grant 443/22 and the Blavatnik fund.
Gil Cohen: Funded by ERC starting grant 949499 and by the Israel Science Foundation grant 2989/24.
Dean Doron: Funded in part by the Israel Science Foundation grant 857/25.
Yuval Khaskelberg: Funded by ERC starting grant 949499.
Amnon Ta-Shma: Funded by the Israel Science Foundation grant 443/22.

Acknowledgements We thank William Hoza for bringing to our attention Lemma 4.9 in [10], which already proves one of the theorems stated in an earlier version of this paper. We also thank Kuan Cheng and Ruiyang Wu for a useful discussion about their work [12].



© Ben Chen, Gil Cohen, Dean Doron, Yuval Khaskelberg, and Amnon Ta-Shma;
licensed under Creative Commons License CC-BY 4.0



1 Introduction

The question of whether $\mathbf{BPL} = \mathbf{L}$ is one of the central open problems in computational complexity theory [1, 3, 26]. It asks whether randomness provides additional computational power to space-bounded algorithms, or equivalently, whether every probabilistic logspace algorithm can be fully derandomized with only a constant-factor increase in space. Despite decades of intensive study, the problem remains open. Nonetheless, the prevailing belief is that $\mathbf{BPL} = \mathbf{L}$, a conjecture supported by plausible circuit lower bounds [28, 20, 19]. Importantly, unlike in the time-bounded setting, there are currently no known barriers that rule out unconditional derandomization of **BPL**.

Progress toward this goal has been strikingly limited. Beyond a small number of landmark results, there has been essentially no improvement for several decades. Already in the work of Savitch [40] and Borodin, Cook, and Pippenger [3], it was shown that $\mathbf{BPL} \subseteq \mathbf{L}^2$. The only substantial strengthening of this bound was obtained by Saks and Zhou [39], who proved that $\mathbf{BPL} \subseteq \mathbf{L}^{3/2}$. This result, established nearly 30 years ago, has remained essentially unimproved to this day, with only a quite recent sub-polynomial improvement due to Hoza [22].

The primary approach to resolving **BPL** vs. **L** is via the construction of pseudorandom generators (PRGs) for read-once branching programs (BPs), which constitute the standard nonuniform model associated with space-bounded Turing machines. We refer the reader to Definition 5 for a formal definition. For present purposes, it suffices to note that the length of a BP, denoted by n, corresponds to the number of random bits read by the algorithm (and is therefore bounded by its running time), while the width w captures the space usage and is, in fact, exponential in the space bound. For naive derandomization of **BPL** via PRGs, one should consider $w = \mathrm{poly}(n)$.

In a seminal paper, Nisan [33] constructed an explicit ε-error PRG for BPs of length n and width w with seed length $O\big(\log n \cdot \log\big(\frac{nw}{\varepsilon}\big)\big)$. This construction – together with its expander-based variant due to Impagliazzo, Nisan, and Wigderson [25] – remains the state of the art in the general setting. An improvement in the regime $w \gg n$ was obtained by Armoni [2] and was recently simplified in [8], shaving a factor of $\log\Big(\frac{\log w}{\log(n/\varepsilon)}\Big)$ from the seed length.[1]

By contrast, an optimal PRG has seed length $O\big(\log\big(\frac{nw}{\varepsilon}\big)\big)$; if such a generator were also space-efficient, it would immediately imply $\mathbf{BPL} = \mathbf{L}$. Remarkably, despite more than three decades of subsequent work, Nisan's construction has not been improved. In fact, except for the case of $w = 3$ [31], or when additional structural restrictions are imposed on the BPs (see, e.g., [29, 16, 5, 17, 36, 18, 9]), no PRG is known to break the so-called log-squared barrier; that is, to achieve seed length $o(\log^2 n)$ for constant error ε, when $w = \mathrm{poly}(n)$, or even when the width is as small as $w = 4$.

Weighted pseudorandom generators

To bypass this long-standing stalemate, Braverman, Cohen, and Garg [4] introduced a new type of pseudorandom object called a *weighted pseudorandom generator (WPRG)*. A WPRG relaxes the classical definition of a PRG while still enabling the derandomization of two-sided

[1] Armoni's generator uses the classical Nisan–Zuckerman generator [34], that gives optimal seed length in the extreme regime of $n = \mathrm{polylog}(w)$.

error probabilistic algorithms. Specifically, in a WPRG each seed is assigned a weight – which may be positive, negative, and potentially unbounded – and the desired ε-approximation is achieved via a weighted sum rather than a simple average.

In [4], the authors constructed a WPRG with an almost optimal dependence on the error parameter ε, achieving seed length $\widetilde{O}(\log n \cdot \log(nw) + \log(1/\varepsilon))$. While the error parameter is taken to be constant in the most naive PRG-based approach to derandomizing **BPL**, it plays a crucial role in more refined derandomization procedures, most notably the Saks–Zhou algorithm. The latter involves multiple composition steps, necessitating small error and making the dependence on ε crucial. Indeed, Braverman et al. [4] and Chattopadhyay and Liao [6] showed that one can prove $\mathbf{BPL} \subseteq \mathbf{L}^{4/3}$ even without breaking the log-squared barrier. Concretely, they showed that such a derandomization result would follow from the existence of a space-efficient WPRG with seed length $O(\log^2 n + \log(w/\varepsilon))$.

As observed in [4], WPRGs readily yield hitting sets. Consequently, their result led to the first improved construction of hitting sets for BPs since the work of Nisan [33], with a much simplified follow-up construction due to Hoza and Zuckerman [24].

Since then, WPRGs have attracted significant attention. They have inspired simpler and improved constructions [30, 6, 14, 36, 22], culminating in the WPRG of Hoza [22], which removes all double-logarithmic factors, obtaining seed length $O(\log n \cdot \log(nw) + \log(1/\varepsilon))$ and in the very recent state-of-the-art construction of Cheng and Wu [12], which achieves further improvements in the regime $w \gg n$.

These works incorporate ideas from fast Laplacian solvers and were also the basis for significantly improved generators for restricted models [23, 36, 9]. The limitations of these techniques have been investigated in [13]. Developments in WPRGs have led to progress on several long-standing problems. Notably, they were leveraged by Hoza to improve the Saks–Zhou derandomization mentioned above [22], and have also enabled substantial advances on the closely related problem of iterated matrix multiplication [15, 35]. Moreover, the error reduction techniques have recently been used to show that **BPL** is contained in logspace-uniform $\mathbf{AC}^1$ [11].

Braverman, Cohen, and Garg [4] emphasized the role of the error parameter ε, showing that simultaneously improving the seed-length dependence on both ε and w would yield a significant improvement to the Saks–Zhou algorithm, even without breaking the log-squared barrier. Moreover, such an improvement need only be achieved for WPRGs, rather than for standard PRGs. While [4] and follow-up works significantly improved the dependence on ε, progress on the width parameter w has remained much more elusive. To date, the only work in this direction that we are aware of is due to Raz and Reingold [37], which presents an intriguing, albeit seemingly stalled, approach.

Cheng and Hoza [10] proposed an intriguing approach toward breaking the log-squared barrier, showing that seed length $O(\log^{3/2} n)$ can be obtained even without *directly* improving the dependence on either n or w. Instead, such an improvement follows from *simultaneously* improving the dependence of PRGs on two secondary parameters: the error ε and the program's arity $|\Sigma|$. We discuss this result below. This underscores the somewhat surprising importance of arity dependence.

1.1 Our Result

The main result of this paper is a construction of WPRGs that achieves optimal dependence on both ε and Σ, while matching the state-of-the-art bounds in n and w. As in prior work, our WPRG is obtained via an error-reduction procedure.

▶ **Theorem 1** (see also Theorem 21). *There exists an explicit reduction that transforms a PRG for length-n, width-w branching programs with seed length s_0 and error $\varepsilon_0 = 1/\mathrm{poly}(n)$, over any alphabet, into a WPRG with error ε and seed length*

$$s_0 + O\left(\log\frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right)\cdot\log w\right).$$

Instantiating Theorem 1 with Armoni's PRG [2], we obtain the following corollary.

▶ **Corollary 2.** *There exists an explicit WPRG for length-n, width-w branching programs over alphabet Σ, achieving error ε with seed length*

$$\log_2|\Sigma| + O\left(\log\frac{1}{\varepsilon} + \frac{\log^2 n + \log n\cdot\log w}{\max(1,\log\log w - \log\log n)} + \log\log\left(\frac{\log w}{\log n}\right)\cdot\log w\right).$$

Corollary 2 improves upon all known WPRGs for general branching programs, including in the regime $w = n^{O(1)}$, where the seed length simplifies to

$$\log_2|\Sigma| + O\big(\log(1/\varepsilon) + \log^2 n\big).$$

In particular, it matches the recent construction of Cheng and Wu [12] in the parameters n, w, and ε, while improving the dependence on Σ. Although this dependence is not made explicit in [12] (nor in most prior work), it is straightforward to verify that their construction – as well as several other recent WPRGs (particularly [22], which is also state-of-the-art in the regime $w = n^{O(1)}$) – exhibits suboptimal dependence on Σ, typically incurring an additional multiplicative factor of $\frac{\log(1/\varepsilon)}{\log n}$.[2]

By contrast, some earlier WPRG constructions (e.g., [4, 14]), while slightly inferior in overall seed length, achieve optimal dependence on Σ. Thus, the contribution of Corollary 2 is to attain optimal dependence on Σ while matching the best known bounds in all other parameters. Moreover, our construction and its analysis appear to be simpler than those of [12], which rely on "weighted pseudorandom reductions" – a sequence of alphabet-reduction and length-reduction steps combined with different PRGs and WPRGs. Furthermore, in the regime $w = n^{O(1)}$, our construction is significantly simpler, and is comparable in simplicity to [22].

As discussed above, our motivation for achieving optimal dependence on the alphabet size stems from its implications for breaking the log-squared barrier [10]. We provide an alternative proof of this implication that requires truly optimal dependence on the alphabet, namely $\log_2|\Sigma|$, as achieved in Corollary 2 for WPRGs. We prove

▶ **Theorem 3.** *Assume there exists an explicit PRG with seed length*

$$\log_2|\Sigma| + O(\log n\cdot\log(nw) + \log(1/\varepsilon)).$$

Then there exists an explicit PRG with seed length[3]

$$\log_2|\Sigma| + O\left(\log^{3/2} n + \log n\cdot\log w + \sqrt{\log n}\cdot\log(1/\varepsilon)\right).$$

---

[2] Specifically, the recent work of Cheng and Wu incurs an additional multiplicative factor of $\frac{\log(n/\varepsilon)}{\log(nw)}$. See [12, Theorem 3.2] in the latest arXiv revision. We also note that in the $n = \mathrm{polylog}(w)$ regime, there exists an $2^{-\log^{0.99} w} \ll \varepsilon \ll 1/\mathrm{poly}(w)$ for which the [12] construction achieves a quadruple-log savings over our construction.

[3] Using error reduction, one can obtain linear dependence on $\log(1/\varepsilon)$ at the cost of producing a WPRG rather than a PRG (see the full version for details [7]).

For the complete details, see [7, Section 5]. The Cheng–Hoza reduction (Lemma 4.9 in [10]) is a recursive construction that uses randomness samplers at each step. Our construction is also recursive, but uses only the hypothesized PRG. We also note that compared to Theorem 3, the applies even when the hypothesized PRG has only $O(\log|\Sigma|)$ dependence on the alphabet size.

2 Proof Overview

In this section, we provide an informal overview of Theorem 1, and outline our alternative proof of the Cheng–Hoza result as given in Theorem 3. The reader may skip this part and proceed directly to the formal sections of the paper. Recall that this theorem takes a PRG for length-n, width-w BPs with error $\varepsilon_0 = 1/n$ and seed length s_0, and produces a WPRG with error ε and seed length

$$s_0 + O\left(\log\frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right)\cdot\log w\right)$$

in the regime $\frac{1}{\varepsilon} > w > n$. When $w < n$ (and more generally, for $w = n^{O(1)}$), the more involved step of our construction can be skipped, yielding a seed length $s_0 + O(\log\frac{1}{\varepsilon})$, and resulting in a significantly simpler construction.

The first component we need for proving Theorem 1 is a variant of the Richardson iteration, which we now turn to discuss.

2.1 Incremental and asymmetric Richardson iteration

The transformation underlying Theorem 1 uses the Richardson iteration, which is by now the standard tool for reducing the error of a PRG. However, our use of the Richardson iteration is more refined, and we begin by briefly describing it here (see Section 4.1 for the formal and complete treatment). For our purposes, the Richardson iteration – or rather, the variant of it used here – is an algorithm Rich that takes as input two WPRGs W_L and W_R for the same length n and width w, with corresponding errors ε_L and ε_R, and produces a new WPRG, denoted

$$\mathrm{Rich}(W_L, W_R),$$

with error guarantee $\varepsilon_L\varepsilon_R$ and seed length

$$s_L + O\left(\log\frac{1}{\varepsilon_L\varepsilon_R}\right).$$

In fact, both the error guarantee and the seed length are slightly more intricate (see Lemma 15), but for the purpose of this overview, it is best to suppress the exact dependence.

How our use of Richardson iteration differs. There are several differences between our implementation of the Richardson iteration and those in the literature. While these distinctions provide useful context, the following discussion is primarily intended for experts; the reader may proceed with the proof overview without focusing on these differences.

In a nutshell, prior works apply the Richardson iteration to a single PRG using several correlated samples, taken with both positive and negative weights, to achieve error cancellation and thus obtain a lower error. The correlated seeds in these works are generated using an auxiliary PRG for BPs. In contrast, our approach reduces the error of two PRGs (and

even WPRGs) which may be imbalanced in their error guarantees and seed lengths. The correlated seeds are obtained via extractors rather than PRGs, allowing for finer control over the seed length of the reduction, which is crucial in the presence of asymmetric WPRGs.

A second difference is our incremental application of the Richardson iteration: rather than applying it in a "one-shot" manner to achieve the desired error, we observe that it is preferable to invoke Richardson on just two instances and then compose the resulting procedure sufficiently many times. This approach is more economical, as the correlated seeds need not meet the final error guarantee from the start. Instead, the early iterations can operate with more coarsely correlated seeds.

Getting back to the proof overview of Theorem 1, the transformation proceeds in two phases:

- Phase 1: Reduce the error from $1/n$ to $1/w$.[4]
- Phase 2. Reduce the error from $1/w$ to the desired error ε.

2.2 Phase 1 – amplifying from inverse-length to inverse-width error

For the first phase – which is the more involved part, and can be skipped in the regime $w = n^{O(1)}$, yielding a simpler construction. The key technical transformation is summarized in the following proposition (see Proposition 19 for the formal statement).

▶ **Proposition 4** (informal). *For any integer $t \geq 1$, a WPRG with error guarantee $\varepsilon_0 < 1/n$ and seed length s_0 can be transformed into a WPRG with error $\varepsilon = \varepsilon_0^t$ and seed length*

$$s_0 + O\left(t^2 \cdot \log \frac{1}{\varepsilon_0} + \log w\right).$$

To be more precise, we assume a bound on the weights of the given WPRG as a function of the other parameters, in particular the error guarantee. This invariant, which we call *error-dominated* (see Definition 16), is preserved throughout our transformations, including the one given in Proposition 4. This ensures that the weights remain under control without the need to track them explicitly. In this informal overview, we will ignore them altogether.

The comb reduction. The proof of Proposition 4 proceeds by applying the Richardson iteration recursively in a "comb-like" manner, as depicted in Figure 1 (which may explain the name).[5] More formally, given a WPRG W_0 with error ε_0 and seed length s_0, we define a sequence of WPRGs $W_1, \ldots, W_{t-1}$ by setting, for each $j \geq 1$,

$$W_j = \mathrm{Rich}\left(W_{j-1}, W_0\right).$$

[4] Strictly speaking, we start with a slightly smaller error, e.g., $1/n^c$ for some constant $c > 1$. We ignore this technicality here.

[5] Using a comb-like construction rather than a balanced binary tree allows us to exploit the asymmetry of the Richardson iteration described above. The "left loss" is the extractor's seed length, whereas the "right loss" combines the extractor's entropy loss with the entropy lost by revealing the program's intermediate state. When $w \gg n$, these losses differ substantially, and the comb lets the growing seed on the left absorb the larger loss on the right. A similar balancing appears already in [8].

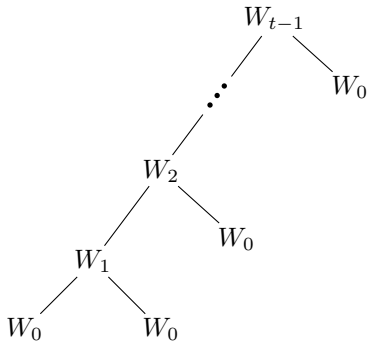


Figure 1 Recursive "comb" structure.

It is easy to see that for each $j \geq 1$, the WPRG W_j has error guarantee $\varepsilon_j = \varepsilon_0^{j+1}$. Consequently, the seed length s_j satisfies the recurrence

$$s_j = s_{j-1} + O\left(\log \frac{1}{\varepsilon_0^{j+1}}\right),$$

which readily yields the seed length bound stated in Proposition 4. The additive $\log w$ term arises from a technical consideration that we suppress in this informal discussion.

Comb of combs. Applying Proposition 4 with $t = \frac{\log w}{\log(1/\varepsilon_0)}$ yields a WPRG with the desired error ε, albeit with a seed length that has a *quadratic* dependence on $\log w$. To understand how this can be avoided, we first ask what error we can "buy" with an additive $O(\log w)$ term in the seed – a cost we are willing to incur. Since the seed length in Proposition 4 is $s_0 + O(t^2 \cdot \log(1/\varepsilon_0))$, we can set

$$t = t_1 = \sqrt{\frac{\log w}{\log(1/\varepsilon_0)}},$$

thereby increasing the seed length by an additive $O(\log w)$ term over s_0. The resulting error guaranteed by Proposition 4 is then $\varepsilon_1 = \varepsilon_0^{t_1}$. Let us (re)denote the resulting WPRG by W_1, which has seed length $s_1 = s_0 + O(\log w)$.

Generally, the error ε_1 is not sufficiently small, and so we repeat the process and construct a new WPRG $W_2 = \mathsf{Comb}_{t_2}(W_1)$ (hence the name "comb-of-combs"; see Figure 2 for an illustration.). Again, we wish to incur only an additive $O(\log w)$ increase in the seed length. Thus, we choose t_2 so that $t_2^2 \cdot \log(1/\varepsilon_1) = \log w$, which yields

$$t_2 = \left(\frac{\log w}{\log(1/\varepsilon_0)}\right)^{1/4} = \sqrt{t}.$$

The resulting error is then $\varepsilon_2 = \varepsilon_0^{t^{1+\frac{1}{2}}}$. Continuing in a recursive manner, we define $W_j = \mathsf{Comb}_{t_j}(W_{j-1})$, with $t_j = \sqrt{t_{j-1}}$. This results in error

$$\varepsilon_j = \varepsilon_0^{t^{1+\frac{1}{2}+\frac{1}{4}+\cdots+\frac{1}{2^{j-1}}}}$$

using seed length $s_0 + O(j \cdot \log w)$. Assuming now that $\varepsilon_0 = 1/n$, a calculation shows that for $\ell = \log\log\left(\frac{\log w}{\log n}\right)$, the WPRG W_ℓ achieves error $1/w$ and seed length $s_0 + O(\ell \cdot \log w)$, which is nearly linear in $\log w$, since ℓ is an extremely slowly growing function.

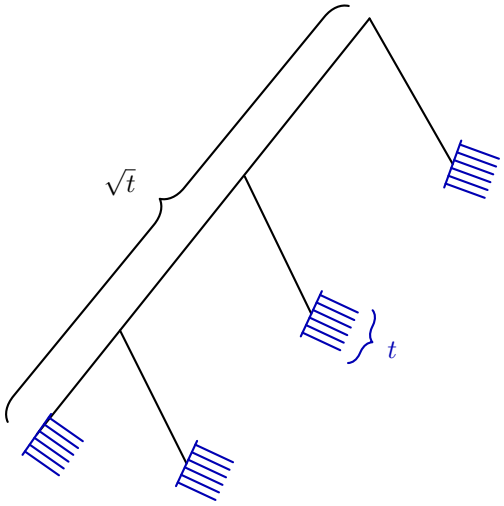


Figure 2 One level illustration of the comb of combs reduction.

2.3 Phase 2 – amplifying from inverse-width error

Once the error is below $1/w$, or if we are already in the regime $w = n^{O(1)}$, there is no further benefit to using asymmetric Richardson iteration, although the incremental approach remains advantageous. To obtain the final WPRG, we therefore compose the Richardson iterations in a binary-tree-like fashion, with the initial WPRG placed at the leaves. More precisely, let W_0 denote the WPRG obtained in the previous sections, with error guarantee $1/w$ and seed length s_0. We then define, for $j = 1, \ldots, h = \log\left(\frac{\log(1/\varepsilon)}{\log w}\right)$,

$$W_j = \mathrm{Rich}(W_{j-1}, W_{j-1}).$$

It is easy to verify that W_h achieves the desired error guarantee ε and has seed length

$$s_0 + 2^h \log w = s_0 + O(\log(1/\varepsilon)).$$

To wrap up, starting with a PRG with error $1/n$ and seed length s_0, the overall seed length, accounting for both phases, is

$$s_0 + O\left(\log \frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right) \cdot \log w\right),$$

as claimed in Theorem 1. Moreover, note that in the regime $w = n^{O(1)}$, only the second phase is invoked, yielding a seed length of $s_0 + O(\log \frac{1}{\varepsilon})$.

2.4 Improving Nisan's PRG via optimal alphabet and error dependence

In this section, we give an overview of our alternative proof of the Cheng–Hoza reduction, as given in Theorem 3, showing that optimal dependence on the alphabet and error suffices to break the log-squared barrier. More precisely, recall that the goal here is to break the log-squared dependence on the length n in Nisan's PRG, assuming the existence of a PRG with optimal alphabet and error dependence. Concretely, suppose we are given a PRG with seed length

$$\log|\Sigma| + O\left(\log^2 n + \log n \cdot \log w + \log \frac{1}{\varepsilon}\right).$$

Our goal in proving Theorem 3 is to show that there exists an efficient transformation which then yields a PRG with seed length

$$\log|\Sigma| + O\left(\log^{3/2} n + \log n \cdot \log w + \sqrt{\log n} \cdot \log \frac{1}{\varepsilon}\right).$$

As we bear no news regarding the dependence on the width w, indeed, our construction exhibits the same $\log n \cdot \log w$ dependence as all previous constructions, we ignore the width parameter for simplicity.[6] Under this simplification, we assume the existence of a PRG with seed length $\log|\Sigma| + O(\log^2 n + \log \frac{1}{\varepsilon})$, and aim to construct a PRG with seed length $\log|\Sigma| + O(\log^{3/2} n + \sqrt{\log n} \cdot \log \frac{1}{\varepsilon})$. The idea is to exploit the optimal dependence on Σ, as follows.

The PRG is constructed recursively with respect to the length n. Fix a parameter r (for the impatient reader, we note that we will take $r = \sqrt{\log n}$), and let $R = 2^r$ and $t = \frac{\log n}{r}$. We construct a sequence of PRGs $G_0, G_1, \ldots, G_t$, all for the same width w and alphabet Σ, where

$$G_j \colon \{0,1\}^{d(j)} \to \Sigma^{R^j}.$$

Note that the final PRG G_t is for length-n BPs, and is the PRG produced by the transformation.

The first PRG, G_0, is simply the identity function, so $d(0) = \log_2|\Sigma|$. For $j \geq 1$ we define G_j by composing R concatenated copies of G_{j-1}. The naive approach would be to define

$$G_j(z) = G_{j-1}(z_1) \circ G_{j-1}(z_2) \circ \cdots \circ G_{j-1}(z_R),$$

where the seed z is partitioned into R disjoint blocks. But this results in an unacceptable blow-up in the seed length, $d(j) = R \cdot d(j-1)$, which ultimately yields a trivial PRG.

Instead, we use the hypothesized PRG to recycle the seeds of the R calls to G_{j-1}. In particular, let H_j denote the hypothesized PRG used to construct G_j. It has the form

$$H_j \colon \{0,1\}^{d(j)} \to \Sigma_j^R,$$

where $\Sigma_j = \{0,1\}^{d(j-1)}$. With H_j in place, we define G_j via "sparsified folding":[7]

$$G_j(z) = G_{j-1}\left(H_j(z)_1\right) \circ G_{j-1}\left(H_j(z)_2\right) \circ \cdots \circ G_{j-1}\left(H_j(z)_R\right).$$

Note that H_j is a PRG with very large alphabet size, exponential in the seed length of G_{j-1}, yet its output length is only R, which we will take to be much smaller. It is at this point that we capitalize on the excellent dependence on the alphabet size. Indeed, by our hypothesis, the seed length of H_j satisfies

$$d(j) = \underbrace{\log|\Sigma_j|}_{d(j-1)} + O\left(r^2 + \log\frac{1}{\varepsilon_H}\right),$$

where ε_H is the error parameter of H_j (taken uniformly across the j-s). Solving the recurrence yields

$$d(t) = O\left(t \cdot \left(r^2 + \log\frac{1}{\varepsilon_H}\right)\right) + \log|\Sigma|.$$

[6] Alternatively, the reader may think of the width w as a constant; recall that breaking the log-squared barrier remains open even for width $w = 4$.

[7] A similar length reduction step is also used in [12].

In the full version [7], we prove that correlating the R seeds using a BP does indeed work, yielding a final error ε bounded by $n\varepsilon_H$. This, together with our choice $t = \frac{\log n}{r}$ yields that the seed length is

$$d(t) = O\left(\frac{\log n}{r} \cdot \left(r^2 + \log n + \log \frac{1}{\varepsilon}\right)\right) + \log|\Sigma|.$$

Optimizing by setting $r = \sqrt{\log n}$, we get the desired seed length.

3 Preliminaries

All logarithms in this paper are taken to base 2, unless otherwise stated. Whenever we write $x \sim A$ for A being a set, we mean that x is sampled uniformly at random from the flat distribution over A. We denote by U_n the random variable distributed uniformly over $\{0,1\}^n$.

We say $f\colon \Lambda_1 \to \Lambda_2^\star$ is explicit if it's computable in time $\mathrm{poly}(\log|\Lambda_1|)$. We mostly do not keep precise track of the space complexity of our error-reduction procedure. A suitable bound follows from standard results on the space complexity of compositions, together with the space complexity of the underlying PRG whose error is being reduced.

3.1 Matrices, Branching Programs, and Space Complexity

A matrix is Boolean if all its entries are in $\{0,1\}$, and stochastic if all its entries are nonnegative and the sum of each column is 1. Denote by $\mathrm{BSto}(w)$ the set of $w \times w$ boolean stochastic matrices. We will denote by $\|\cdot\|$ the induced ℓ_1 norm, i.e., $\|A\| = \max_j \sum_i |A_{i,j}|$.

We start by defining standard-order, read-once, branching programs.

▶ **Definition 5** (branching program). *Let Σ be some alphabet and let $n, w \in \mathbb{N}$. An (n, w, Σ) branching program (BP) is a sequence $B = (B_1, \ldots, B_n)$, where each $B_i\colon \Sigma \to \mathrm{BSto}(w)$.*

For $i \le j$ we let $B_{i\to j}$ be the $(j-i+1, \Sigma, w)$ BP $(B_i, \ldots, B_j)$.

▶ **Definition 6.** *The* value *of an (n, w, Σ) BP $B = (B_1, \ldots, B_n)$ on $x = (x_1, \ldots, x_n) \in \Sigma^n$, denoted $B(x)$, is the realized $w \times w$ matrix of B when fed by x, i.e.,*

$$B(x) = B_n(x_n) \cdot B_{n-1}(x_{n-1}) \cdots B_1(x_1).$$

If B is the empty sequence, we set $\emptyset(x) = I_w$.

▶ **Definition 7** (weighted PRG). *We say (G, μ) is an $(n, w, \Sigma, \Gamma, \varepsilon)$-WPRG against BPs with seed length s if:*

- *$G\colon \{0,1\}^s \to \Sigma^n$ and $\mu\colon \{0,1\}^s \to \mathbb{R}$,*
- *For every (n, w, Σ) BP $B = (B_1, \ldots, B_n)$, it holds that*

$$\left\|\mathbb{E}_{x\sim\{0,1\}^s}[\mu(x) \cdot B(G(x))] - \mathbb{E}_{x\sim\Sigma^n}[B(x)]\right\| \le \varepsilon,$$

- *For every $x \in \{0,1\}^s$, $|\mu(x)| \le \Gamma$.*

When $\mu \equiv 1$, the notion coincides with that of a PRG, in which case we omit the Γ from the tuple.

For $1 \le i \le j \le n$ we let $G_{i\to j}\colon \{0,1\}^s \to \Sigma^{j-i+1}$ be the corresponding truncation of G's output. Note that if (G, μ) is an $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG, then $(G_{i\to j}, \mu)$ is an $(j-i+1, w, \Sigma, \Gamma, \varepsilon)$ WPRG. We will often omit the subscript $i \to j$ when the length of the corresponding BP is clear from context.

3.2 Known PRG Constructions

▶ **Theorem 8** ([25]). *For any positive integers n, w, any error parameter $\varepsilon > 0$ and any alphabet Σ, there exists an explicit $(n, \Sigma, w, \varepsilon)$ PRG with seed length*

$$s = \log|\Sigma| + O\Big(\log n \cdot \log\Big(\frac{nw}{\varepsilon}\Big)\Big).$$

When $w \gg n$, Armoni's PRG [2] improves upon the INW generator. When instantiated with modern extractors, it works for all ranges of parameters [27], and has the optimal dependence on Σ [8].

▶ **Theorem 9** ([2, 27, 8]). *For any positive integers n, w, any error parameter $\varepsilon > 0$ and any alphabet Σ, there exists an explicit $(n, \Sigma, w, \varepsilon)$ PRG with seed length*

$$s = \log|\Sigma| + O\left(\frac{\log(nw/\varepsilon)\log n}{\max(1, \log\log w - \log\log(n/\varepsilon))}\right).$$

3.3 Seeded Extractors

The *support* of a random variable X distributed over some domain Ω is the set $x \in \Omega$ for which $\Pr[X = x] \neq 0$, which we denote by $\mathrm{Supp}(X)$. The *total variation distance* (or, statistical distance) between two random variables X and Y over the same domain Ω is defined as $|X - Y| = \max_{A \subseteq \Omega}(\Pr[X \in A] - \Pr[Y \in A])$. Whenever $|X - Y| \leq \varepsilon$ we say that X is ε-close to Y and denote it by $X \approx_\varepsilon Y$.

For a function $f\colon \Omega_1 \to \Omega_2$ and a random variable X distributed over Ω_1, $f(X)$ is the random variable distributed over Ω_2 obtained by choosing x according to X and computing $f(x)$. For a set $A \subseteq \Omega_1$, $f(A) = \{f(x) : x \in A\}$. For every $f\colon \Omega_1 \to \Omega_2$ and two random variables X and Y distributed over Ω_1 it holds that $|f(X) - f(Y)| \leq |X - Y|$, and this is often referred to as the data-processing inequality.

▶ **Definition 10** (min-entropy). *Let X be a discrete random variable over a set $\mathcal{X}$. The* min-entropy *of X, denoted by $H_\infty(X)$, is defined as*

$$H_\infty(X) = \min_{x \in \mathcal{X}}(-\log\Pr[X = x]) = -\log\left(\max_{x \in \mathcal{X}}\Pr[X = x]\right). \tag{1}$$

▶ **Definition 11** ((n,k) source). *An (n,k)* source *is a random variable X distributed over $\{0,1\}^n$ such that its min-entropy is at least k, i.e., $H_\infty(X) \geq k$. Equivalently, it is a distribution where for all $x \in \{0,1\}^n$,*

$$\Pr[X = x] \leq 2^{-k}. \tag{2}$$

▶ **Definition 12** (extractor). *A function $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k,ε) (seeded) extractor if the following holds. For every (n,k) source X it holds that $\mathsf{Ext}(X, Y) \approx_\varepsilon U_m$, where Y is uniformly distributed over $\{0,1\}^d$ and is independent of X.*

We will use two extractors, one in the balanced regime, and one in the unbalanced regime.

▶ **Theorem 13** ([21]). *There exists a constant $c_{\mathsf{seed}} \geq 1$ such that the following holds. For every positive integer n, and any $\Delta < n$ and $\varepsilon > 0$, there exists an explicit $(k = n - \Delta, \varepsilon)$ extractor $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^n$, where $d = c_{\mathsf{seed}} \cdot (\Delta + \log(n/\varepsilon))$.*

▶ **Theorem 14** ([38, 6]). *There exist constants $c_{\mathsf{seed}}, c_{\mathsf{loss}} \geq 1$ such that the following holds. For every positive integer n, and any $\Delta < n$ and $\varepsilon > 0$, there exists an explicit $(k = n - \Delta, \varepsilon)$ extractor $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, where $d = c_{\mathsf{seed}} \cdot \log(\Delta/\varepsilon)$ and $m = n - c_{\mathsf{loss}} \cdot (\Delta + \log\frac{1}{\varepsilon})$.*

4 Our Error Reduction Procedure

In this section we prove Theorem 1 and deduce Corollary 2. We begin with an informal overview of the Richardson iteration technique. This procedure depends on a parameter k that governs the quality of error reduction: roughly, an error-ε PRG is transformed into an error-ε^k WPRG. As discussed in the proof overview, in this work we observe that it is more economical to work with $k = 2$ and then compose the resulting WPRGs.

Our use of the Richardson iteration is asymmetric. In particular, in our error reduction we apply the iteration to WPRGs with varying error parameters. To set the stage, we first review, at an informal level, the Richardson iteration with parameter $k = 2$ in the symmetric setting. With this intuition in place, we then proceed in Section 4.1 to formally construct the Richardson iteration with parameter $k = 2$ for the asymmetric case.

Suppose we are given

- $w \times w$ sub-stochastic matrices $M_1, \ldots, M_n$, and,
- Crude approximations $\widetilde{M}_{i,j}$ to $M_i M_{i-1} \cdots M_j$ for all $1 \le j < i \le n$ (for $j = i$ we naturally take $\widetilde{M}_{i,i} = M_i$).

In the context of space bounded derandomization, [32] describe a method to use $M_1, \ldots, M_n$ to refine the crude approximations to much better approximations. First, they "plant" the matrix product problem within a matrix inversion problem. Specifically, they form the $(n+1) \times (n+1)$ block matrix

$$M = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ M_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & M_2 & 0 & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 & 0 \\ 0 & 0 & \cdots & M_{n-1} & 0 & 0 \\ 0 & 0 & \cdots & 0 & M_n & 0 \end{pmatrix} \tag{3}$$

and define $L = I - M$. Then they observe that the inverse of L is:

$$(L^{-1})_{i,j} = \begin{cases} M_{i-1} \cdots M_j & \text{if } i > j, \\ I & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, the product we wish to approximate is a block within L^{-1}. We can use the crude approximations $\widetilde{M}_{i-1,j}$ to form the matrix

$$\widetilde{L^{-1}}_{i,j} = \begin{cases} \widetilde{M}_{i-1,j}, & \text{if } i > j, \\ I, & \text{if } i = j, \\ 0, & \text{if } i < j. \end{cases}$$

and $\widetilde{L^{-1}}$ is a crude approximation to L^{-1}. Prior works apply Richardson iteration to get a better approximation to L^{-1}. Specifically they take

$$\widehat{\widetilde{L^{-1}}} = 2\widetilde{L^{-1}} - \widetilde{L^{-1}} L \widetilde{L^{-1}}, \tag{4}$$

and it is immediate to check that

$$\left\| I - L\widehat{\widetilde{L^{-1}}} \right\| = \left\| I - L\widetilde{L^{-1}} \right\|^2.$$

Thus, $\widehat{\widetilde{L^{-1}}}$ yields a quadratically improved approximation of L^{-1} and therefore also of $M_1 \cdot \ldots \cdot M_n$. An explicit formula for the approximation of $M_1 \cdot \ldots \cdot M_n$ can be derived: For every $n \geq 3$,

$$\widehat{\widetilde{M}}_{n+1,1} = \sum_{r=1}^{n-2} \widetilde{M}_{n,r+2} M_{r+1} \widetilde{M}_{r,1} - \sum_{r=1}^{n-3} \widetilde{M}_{n,r+2} \widetilde{M}_{r+1,1}. \tag{5}$$

For example, for $n = 4$, the approximation of $M_4 M_3 M_2 M_1$ is

$$\widehat{\widetilde{M}}_{5,1} = \widetilde{M}_{4,3} M_2 M_1 + M_4 M_3 \widetilde{M}_{2,1} - \widetilde{M}_{4,3} \widetilde{M}_{2,1}.$$

Observe that Equation (5) is expressed as the difference of two sums. In the first (positive) sum, each summand is a product of three terms – two given approximations with an input matrix M_r sandwiched in the middle. In the second (negative) sum, the product omits this middle input matrix.

[36] and [14] independently notice that if the approximations $\widetilde{M}$ are obtained with a PRG, then the approximation $\widehat{\widetilde{M}}$ can be realized with a WPRG [8]. A naive (and expensive) implementation of this would construct the WPRG for $\widehat{\widetilde{M}}$ using two independent calls to the WPRGs for $\widetilde{M}$. A better idea is to recycle the seeds needed for the two calls to the WPRG of $\widetilde{M}$. In this work we explore what happens when we employ two different WPRGs, each with a different accuracy level, and we show that this leads to better WPRGs.

4.1 Asymmetric Richardson Iteration

In this section we make the above idea precise.

Construction. We are given two WPRGs, which we think of as a "left" WPRG and a "right" one, possibly with different parameters, and an extractor.

- An $(n, w, \Sigma, \Gamma_L, \varepsilon_L)$ WPRG (G^L, μ_L) with seed length s_L.
- An $(n, w, \Sigma, \Gamma_R, \varepsilon_R)$ WPRG (G^R, μ_R) with seed length s_R.
- A $(k, \varepsilon_{\mathsf{Ext}})$ extractor $\mathsf{Ext}\colon \{0,1\}^{s_L} \times \{0,1\}^{d_{\mathsf{Ext}}} \to \{0,1\}^{s_R}$, where $k = s_L - \log \frac{w}{\varepsilon_{\mathsf{Ext}}}$.

The construction of our new WPRG

$$\mathrm{Rich}_{\mathsf{Ext}}((G^L, \mu_L), (G^R, \mu_R)) = (G, \mu)$$

goes as follows. We interpret the set of seeds S as $[2n-5] \times \{0,1\}^{s_L} \times \{0,1\}^{d_{\mathsf{Ext}}} \times \{0,1\}^{d_{\mathsf{Ext}}}$. Given $(t, x_L, y_1, y_2) \in S$, we set $t' = t - n + 2$, and define

$$G(t, x_L, y_1, y_2) = \begin{cases} G^L_{1\to t}(x_L) \circ \mathsf{Ext}(x_L, y_2)_{[1,\log|\Sigma|]} \circ G^R_{t+2\to n}(\mathsf{Ext}(x_L, y_1)) & \text{if } t \leq n-2, \\ G^L_{1\to t'+1}(x_L) \circ G^R_{t'+2\to n}(\mathsf{Ext}(x_L, y_1)) & \text{otherwise.} \end{cases}$$

$$\mu(t, x_L, y_1, y_2) = (2n-5) \cdot \begin{cases} \mu_L(x_L) \cdot \mu_R(\mathsf{Ext}(x_L, y_1)) & \text{if } t \leq n-2, \\ -\mu_L(x_L) \cdot \mu_R(\mathsf{Ext}(x_L, y_1)) & \text{otherwise.} \end{cases}$$

Note that the construction exactly corresponds to Equation (5), where we use two different WPRGs for approximating the matrices. Notice that the first $n-2$ terms come with a positive sign, and the last $n-3$ terms with a negative sign.

[8] The use of weights here is essential, as some of terms in Equation (5) come with a negative weight.

▶ **Lemma 15.** *Given* (G^L, μ_L), (G^R, μ_R), *and* Ext *as above, it holds that* (G, μ) *is an* $(n, w, \Sigma, \Gamma, \varepsilon)$ *WPRG, where:*

- *The seed length* $s = \log|S|$ *satisfies* $s = \log(2n-5) + s_L + 2d_{\mathsf{Ext}}$,
- *The weight of* G *is at most* $\Gamma = c \cdot n\Gamma_L\Gamma_R$ *for some constant* $c \geq 1$, *and,*
- *The error* ε *is bounded by* $\varepsilon \leq 4(n+1)^2\varepsilon_L\varepsilon_R + \Gamma \cdot \varepsilon_{\mathsf{Ext}}$.

Proof. The weighted generator (G, μ) treats each seed as a quadruple (t, x_L, y_1, y_2). Here $t \in [2n-5]$ specifies which of the $(n-2)$ "positive" or $(n-3)$ "negative" summands in the Richardson expansion is selected; $x_L \in \{0,1\}^{s_L}$ is drawn according to μ_L; and $y_1, y_2 \in \{0,1\}^{d_{\mathsf{Ext}}}$ are independent extractor seeds. Encoding t requires $\log(2n-5)$ bits, x_L requires s_L bits, and the two extractor seeds contribute $2\,d_{\mathsf{Ext}}$ bits in total, giving a seed length of

$$s = \log(2n-5) + s_L + 2\,d_{\mathsf{Ext}}.$$

Each output of (G, μ) has weight

$$\mu(t, x_L, y_1, y_2) = \pm(2n-5)\,\mu_L(x_L)\,\mu_R\big(\mathsf{Ext}(x_L, y_1)\big),$$

and therefore satisfies

$$|\mu(t, x_L, y_1, y_2)| \leq (2n-5)\,\Gamma_L\,\Gamma_R.$$

We move to analyzing the error of the weighted PRG.

Constructing approximate inverses. Let B be an (n, w, Σ) BP with transition matrices $M_1, \ldots, M_n$. Let L be the corresponding $(n+1)w \times (n+1)w$ block matrix defined in Equation (3) whose inverse L^{-1} encodes all partial products $M_i \cdots M_{j+1}$. We form two approximate inverses $L^{-1(L)}$ and $L^{-1(R)}$ by replacing each block $M_{i,j}$ with

$$\widetilde{M}^{(L)}_{i,j} = \mathop{\mathbb{E}}_{x \sim \{0,1\}^{s_L}}\Big[\mu_L(x) \cdot B_{i\to j}\big(G^L(x)\big)\Big] \quad \text{or} \quad \widetilde{M}^{(R)}_{i,j} = \mathop{\mathbb{E}}_{x \sim \{0,1\}^{s_R}}\Big[\mu_R(x) \cdot B_{i\to j}\big(G^R(x)\big)\Big],$$

respectively. Recalling that (G^L, μ_L) and (G^R, μ_R) have errors ε_L and ε_R, we have

$$\big\|M_{i,j} - \widetilde{M}^{(L)}_{i,j}\big\| \;\leq\; \varepsilon_L, \quad \big\|M_{i,j} - \widetilde{M}^{(R)}_{i,j}\big\| \;\leq\; \varepsilon_R.$$

Summing these errors over the $n{+}1$ block rows gives

$$\big\|L^{-1} - L^{-1(L)}\big\| \;\leq\; (n{+}1)\,\varepsilon_L, \quad \big\|L^{-1} - L^{-1(R)}\big\| \;\leq\; (n{+}1)\,\varepsilon_R.$$

As $\|L\| \leq 2$,

$$\big\|I - L^{-1(L)}L\big\| \;\leq\; 2(n{+}1)\,\varepsilon_L, \qquad \big\|I - L^{-1(R)}L\big\| \;\leq\; 2(n{+}1)\,\varepsilon_R.$$

Asymmetric Richardson. "Vanilla" Richardson refines a *single* crude inverse $A \approx L^{-1}$ via

$$A \longmapsto 2A - ALA,$$

which satisfies $\|I - (2A - ALA)L\| = \|I - AL\|^2$. In principle, introducing two approximations looks unnecessary–one could just reuse the more accurate one. In our setting, however, the two appearances of $\widetilde{L^{-1}}$ are generated from *different seeds*: the left term is driven by

x_L, while the right term is obtained from x_L through an extractor and therefore may have different accuracy. This asymmetry naturally leads us to use a Richardson-type update that mixes two approximations,

$$R = L^{-1(L)} + L^{-1(R)} - L^{-1(L)} L L^{-1(R)}.$$

A short calculation shows that

$$I - RL = (I - L^{-1(L)}L)(I - L^{-1(R)}L).$$

Consequently,

$$\|I - RL\| \le \|I - L^{-1(L)}L\| \, \|I - L^{-1(R)}L\| \le 4(n+1)^2 \varepsilon_L \varepsilon_R,$$

and in particular the bottom-left block satisfies

$$\||R[n+1,1] - M_n \cdots M_1\| \le 4(n+1)^2 \varepsilon_L \varepsilon_R.$$

Uncorrelated seeds. Assume that the seed x_R for G_R and the symbol $\sigma \in \Sigma$ are sampled independently from x_L and uniformly. Given $t \in [2n-5]$, we consider the above two cases. When $t \in [n-2]$, letting $r = t$, the generator outputs $G^L_{1\to r}(x_L) \circ \sigma \circ G^R_{r+2\to n}(x_R)$. Then,

$$\mathbb{E}_{x_L,x_R,\sigma}\Big[\mu_L(x_L)\,\mu_R(x_R)\,B\big(G^L_{1\to r}(x_L) \circ \sigma \circ G^R_{r+2\to n}(x_R)\big)\Big] = \widetilde{M}^{(R)}_{n,r+2}\, M_{r+1}\, \widetilde{M}^{(L)}_{r,1}.$$

When $t > n-2$, we denote r to be such that $t = n-2+r$, and now

$$\mathbb{E}_{x_L,x_R}\Big[\mu_L(x_L)\,\mu_R(x_R)\,B\big(G^L_{1\to r+1}(x_L) \circ G^R_{r+2\to n}(x_R)\big)\Big] = -\,\widetilde{M}^{(R)}_{n,r+2}\, \widetilde{M}^{(L)}_{r+1,1}.$$

Averaging over $t \in [2n-5]$ and multiplying by $2n-5$ yields exactly

$$\sum_{r=1}^{n-2} \widetilde{M}^{(L)}_{n,r+2}\, M_{r+1}\, \widetilde{M}^{(R)}_{r,1} - \sum_{r=1}^{n-3} \widetilde{M}^{(L)}_{n,r+2}\, \widetilde{M}^{(R)}_{r+1,1},$$

which is the $(n+1,1)$-block of the Richardson matrix R. Thus, when x_R and σ are drawn independently, the weighted generator's output matches $R[n+1,1]$.

Correlating x_R and σ with x_L via two extractor calls. We now replace the independent choices (x_R, σ) by

$$x_R = \mathsf{Ext}(x_L, y_1) \in \{0,1\}^{s_R}, \quad \sigma = \mathsf{Ext}(x_L, y_2)_{[1,\log|\Sigma|]} \in \Sigma,$$

where y_1, y_2 are independent uniform seeds (and independent of x_L). We bound the difference in the ℓ_1 norm of the $(n+1,1)$-block contribution.

We use the standard seed-recycling argument for width-w ROBPs. Conditioning on reaching the state $q \in [w]$ after reading x_L reveals at most, roughly, $\log w$ bits of information; Formally, except with probability at most $\varepsilon_{\mathsf{Ext}}$, this conditioning reduces the *min-entropy* of the left seed by at most $\log(w/\varepsilon_{\mathsf{Ext}})$. Setting $\Delta := \log(w/\varepsilon_{\mathsf{Ext}})$, and using our extractor $\mathsf{Ext} : \{0,1\}^{s_L} \times \{0,1\}^{d_{\mathsf{Ext}}} \to \{0,1\}^{s_R}$ with entropy loss Δ and error $\varepsilon_{\mathsf{Ext}}$, the condition

$$s_L - \Delta \ge s_L - \log \frac{w}{\varepsilon_{\mathsf{Ext}}} = k$$

guarantees that $\mathsf{Ext}(x_L, U)$ is $\varepsilon_{\mathsf{Ext}}$-close to uniform even after conditioning on q. Consequently, for any functions f_L, f_R,

$$\left\|\mathbb{E}_{x_L,y}\big[B\big(f_L(x_L)\circ f_R(\mathsf{Ext}(x_L,y))\big)\big] - \mathbb{E}_{x_L,x_R}\big[B\big(f_L(x_L)\circ f_R(x_R)\big)\big]\right\| \le 2\varepsilon_{\mathsf{Ext}}.$$

Taking the weights into account, recalling that $|\mu_L| \le \Gamma_L$ and $|\mu_R| \le \Gamma_R$, we have

$$\begin{aligned}\Big\|\mathbb{E}_{x_L,y}\big[(2n-5)\mu_L(x_L)\mu_R(\mathsf{Ext}(x_L,y))\cdot B(f_L(x_L)\circ f_R(\mathsf{Ext}(x_L,y)))\big] \\ - \mathbb{E}_{x_L,x_R}\big[(2n-5)\mu_L(x_L)\mu_R(x_R)\cdot B(f_L(x_L)\circ f_R(x_R))\big]\Big\| \le c\cdot(2n-5)\Gamma_L\Gamma_R\varepsilon_{\mathsf{Ext}}\end{aligned}$$

for some constant $c \ge 1$. The ultimate error bound is obtained by the triangle inequality. ◀

To conclude this section, we define a useful invariant which will be helpful towards establishing our WPRG.

▶ **Definition 16.** *We say that an $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG is* error-dominated *if*

$$\Gamma \le \frac{1}{c(n+1)^3\varepsilon},$$

where c is the same constant from Lemma 15.

Note that error-dominated WPRGs necessarily have low error. We record the following observation.

▷ Claim 17. Keeping the notation above, particularly the constant c from Lemma 15, assume that both (G^L, μ_L) and (G^R, μ_R) are error-dominated, and that

$$\varepsilon_{\mathsf{Ext}} \le \frac{n^2\varepsilon_L\varepsilon_R}{c\Gamma}.$$

Then, (G, μ) is error-dominated.

Proof. By Lemma 15, the composed generator satisfies

$$\Gamma = cn\,\Gamma_L\Gamma_R \qquad \text{and} \qquad \varepsilon \le 4(n+1)^2\varepsilon_L\varepsilon_R + \Gamma\varepsilon_{\mathsf{Ext}}.$$

Using $\varepsilon_{\mathsf{Ext}} \le \frac{n\varepsilon_L\varepsilon_R}{c\Gamma}$, we get

$$\varepsilon \le \big(4(n+1)^2 + n^2\big)\varepsilon_L\varepsilon_R \le 5(n+1)^2\varepsilon_L\varepsilon_R.$$

Since (G^L, μ_L) and (G^R, μ_R) are error-dominated,

$$\Gamma_L \le \frac{1}{c(n+1)^3\varepsilon_L} \qquad \text{and} \qquad \Gamma_R \le \frac{1}{c(n+1)^3\varepsilon_R}.$$

Therefore,

$$\Gamma \le cn\Gamma_L\Gamma_R \le \frac{cn}{c^2(n+1)^6}\cdot\frac{1}{\varepsilon_L\varepsilon_R} \le \frac{cn}{c^2(n+1)^6}\cdot\frac{5(n+1)^2}{\varepsilon} = \frac{n}{c(n+1)^4}\cdot\frac{1}{\varepsilon} \le \frac{1}{c(n+1)^3\varepsilon}.\text{◁}$$

4.2 A Comb Reduction

In this section we define a comb-like construction, that composes our asymmetric Richardson construction given in Section 4.1 as follows. For some base (error-dominated) WPRG W_0, and a sequence of extractors $\mathsf{Ext}_1, \ldots, \mathsf{Ext}_t$, we define W_i recursively as

$$W_i = \mathrm{Rich}_{\mathsf{Ext}_i}(W_{i-1}, W_0).$$

We denote by $W_0 = (G_0, \mu_0)$ our $(n, w, \Sigma, \Gamma_0, \varepsilon_0)$ error-dominated WPRG, and by s_0 its seed length. We proceed to set the parameters of our extractors, each of the t extractors

$$\mathsf{Ext}_i \colon \{0,1\}^{s(i)} \times \{0,1\}^{d(i)} \to \{0,1\}^{s_0}$$

is given in Theorem 14 and set to have error $\varepsilon_{\mathsf{Ext}}(i)$, where the $\varepsilon_{\mathsf{Ext}}(i)$'s are as follows. Define, for each $i \in [t]$, $\varepsilon(i) = (5(n+1)^2\varepsilon_0)^i\varepsilon_0$. Note that the sequence of $\varepsilon(i)$-s satisfy $\varepsilon(i) = 5(n+1)^2\varepsilon(i-1)\cdot\varepsilon_0$. For each $i \in [t]$ set

$$\varepsilon_{\mathsf{Ext}}(i) = \frac{n^2\varepsilon_0}{c\Gamma(i)} \cdot \varepsilon(i-1),$$

where $\Gamma(i) = (cn\Gamma_0)^i\Gamma_0$. Note that the $\Gamma(i)$-s satisfy the recurrence relation $\Gamma(i) = cn\Gamma(i-1)\Gamma_0$.

By Lemma 15, the input lengths satisfy $s(1) = s_0 + c_{\mathsf{seed}} \log(w/\varepsilon_{\mathsf{Ext}}(1))$, and for each $2 \le i \le t$,

$$s(i+1) = s(i) + 2c_{\mathsf{seed}} \log(\log w/\varepsilon_{\mathsf{Ext}}(i+1)) + \log(2n-5).$$

We record that:

▷ Claim 18. For every $i \in [t]$ it holds that

$$\log \frac{1}{\varepsilon_{\mathsf{Ext}}(i)} = O\left(i \log \frac{n\Gamma_0}{\varepsilon_0}\right),$$
$$s(i) = s_0 + O\left(\log w + i^2 \log \frac{n\Gamma_0}{\varepsilon_0}\right),$$

and

$$d(i) = O\left(\log\log w + i \log \frac{n\Gamma_0}{\varepsilon_0}\right).$$

Proof. By definition of $\varepsilon_{\mathsf{Ext}}(i)$ and error domination from Claim 17:

$$\varepsilon_{\mathsf{Ext}}(i) = \frac{\varepsilon(i-1)\varepsilon_0 n^2}{c\Gamma(i)} \ge \frac{1}{c} \cdot \varepsilon(i-1)\varepsilon_0 n^2 \cdot c(n+1)^3\varepsilon(i) \ = \frac{(n+1)n^2\varepsilon(i)^2}{5},$$

so

$$\log \frac{1}{\varepsilon_{\mathsf{Ext}}(i)} = O\left(i \cdot \log \frac{n\Gamma_0}{\varepsilon_0}\right).$$

For the WPRG's seed length, we have

$$\begin{aligned} s(i) &= s_0 + s(1) + (i-1)\cdot\log(2n-5) + 2c_{\mathsf{seed}} \sum_{k=2}^{i} \log \frac{\log w}{\varepsilon_{\mathsf{Ext}}(k+1)} \\ &= s_0 + O\left(\log w + i \log\log w + i^2 \log \frac{n\Gamma_0}{\varepsilon_0}\right). \end{aligned}$$

By the construction given in Section 4.1, the entropy loss of each extractor is $\Delta(i) = \log(w/\varepsilon_{\mathsf{Ext}}(i))$. Thus, we can indeed output s_0 bits as

$$s(i) \geq s_0 + c_{\mathsf{loss}}\left(\Delta(i) + \log\frac{1}{\varepsilon_{\mathsf{Ext}}(i)}\right).$$

We thus get that

$$d(i) = O\left(\log\log w + \log\frac{1}{\varepsilon_{\mathsf{Ext}}(i)}\right) = O\left(\log\log w + i\log\frac{n\Gamma_0}{\varepsilon_0}\right). \qquad \lhd$$

We denote this section's transformation, which on input W_0 construct W_t, as $\mathsf{Comb}_t(W_0)$. The parameters of the construction are summarized in the following theorem, whose proof readily follows from Lemma 15, Claim 17, and Claim 18.

▶ **Proposition 19.** *Keeping the above notation, assuming $\varepsilon_0 \leq 1/n^5$ and that $s_0 \geq \log|\Sigma| + \Omega(\log(w/\varepsilon_0))$, $\mathsf{Comb}_t(W_0)$ is an error-dominated $(n, w, \Sigma, \Gamma_t, \varepsilon_t)$ WPRG with seed length*

$$s_0 + O\left(t^2\log\frac{1}{\varepsilon_0} + \log w\right),$$

and error $\varepsilon_t = \varepsilon_0^{t/2}$.

We emphasize that error domination of the WPRG is preserved, and moreover, that if W_0 is explicit, W_t is explicit as well.

4.3 Amplifying from Inverse-Length to Inverse-Width Error

To reduce the error from $n^{-\Omega(1)}$ to $w^{-\Omega(1)}$, we define a "comb-of-combs" construction as follows. Letting W_0 be an error dominated $(n, w, \Sigma, \Gamma_0, \varepsilon_0)$ WPRG that satisfies the assumptions of Proposition 19, for some $t_1, \ldots, t_\ell$, we define the sequence of WPRGs defined by

$$W_i = \mathsf{Comb}_{t_i}(W_{i-1}),$$

and set $\mathsf{CoC}_{t_1,\ldots,t_\ell}(W_0) = W_\ell$.

Concretely, we start with $\varepsilon_0 = 1/n^5$, choose

$$\ell = \log\log\left(\frac{\log w}{\log n}\right),$$

and for each $i \in [\ell]$, we take

$$t_i = 2\left(\frac{\log w}{\log n}\right)^{2^{-i}}.$$

Under this choice of parameters, $\mathsf{CoC}_{t_1,\ldots,t_\ell}(W_0)$ satisfies the following.

▶ **Proposition 20.** *Keeping the above notation,*

$$\mathsf{CoC}_{t_1,\ldots,t_\ell}(W_0)$$

is an explicit error-dominated $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG with $\varepsilon = 1/w$, seed length $s_0 + O(\ell\log w)$.

Proof. First, note that the fact that W_ℓ is error dominated already follows from Proposition 19, and we proceed to analyzing the error. Letting ε_i be the error of W_i, by Proposition 19 we know that $\varepsilon_i = \varepsilon_{i-1}^{t_i/2}$. By our choice of t_i-s, denoting $r = \frac{\log w}{\log n}$, we get that

$$\varepsilon_\ell = \varepsilon_0^{2^{-\ell}\prod_{i\in[\ell]} t_i} = \varepsilon_0^{\prod_{i\in[\ell]} r^{2^{-i}}} = \varepsilon_0^{r^{1-2^{-\ell}}}.$$

Now, $2^\ell = \log r$, so $r^{1-2^{-\ell}} = r/2$. Recalling that $\varepsilon_0 = n^{-5}$, we get $\varepsilon_0^{r/2} = n^{-5r/2} \le 1/w$.

For the seed length, denote by s_i the seed length of W_i. By Proposition 19, we know that each $s_i = s_{i-1} + O(t_i^2 \log(1/\varepsilon_{i-1}))$. Fixing some $i \in [\ell]$, note from the above that

$$t_i^2 \cdot \log\frac{1}{\varepsilon_{i-1}} = t_i^2 \cdot r^{1-2^{-(i-1)}} \log\frac{1}{\varepsilon_0} = 4r^{2^{-i+1}} \cdot r^{1-2^{-i+1}} \log\frac{1}{\varepsilon_0} = 20\log n \cdot r = 20\log w.$$

Thus, $s_i = s_{i-1} + O(\log w)$ and we're done with the seed length caclculation. Finally, note that in order to invoke Proposition 19, we need to make sure that $s_i \ge \log|\Sigma| + \Omega(\log(w/\varepsilon_i))$, which follows from the fact that $t_i^2 \log(1/\varepsilon_{i-1}) = \Omega(\log(1/\varepsilon_i))$ and that $s_0 \ge \log|\Sigma| + \Omega(\log w)$. ◀

4.4 Amplifying to an Arbitrary Error

Once we achieved an error of $1/w$, we reduce the error to an arbitrary desired error ε by recursively applying the Richardson iteration on the same WPRG, i.e., at the base of the recursion we use CoC above on the given base-WPRG W. More formally, we start with $V_0 = \mathsf{CoC}_{t_1,\ldots,t_\ell}(W)$ as above, set

$$h = \log\left(\frac{\log(1/\varepsilon)}{\log(w/6n^2)}\right),$$

and for each $i \in [h]$, define

$$V_i = \mathrm{Rich}_{\mathsf{Ext}_i}(V_{i-1}, V_{i-1}),$$

where the extractors $\mathsf{Ext}_1, \ldots, \mathsf{Ext}_h$ are defined as follows. We denote the seed length of each $(n, w, \Sigma, \Gamma_i, \varepsilon_i)$ WPRG V_i by s_i, and the seed length of W by $s(W)$. Each

$$\mathsf{Ext}_i \colon \{0,1\}^{s_{i-1}} \times \{0,1\}^{d_i} \to \{0,1\}^{s_{i-1}}$$

is taken to be the $(k_i, \varepsilon_{\mathsf{Ext}_i})$ extractor of Theorem 13 as in Section 4.2, where $k_i = s_{i-1} - \log(w/\varepsilon_{\mathsf{Ext}_i})$, and $\varepsilon_{\mathsf{Ext}_i} = \frac{n^2\varepsilon_{i-1}^2}{c\Gamma_i}$. By Lemma 15, we have the recurrence relation relating the seed lengths of the WPRGs

$$s_i = s_{i-1} + 2d_i + O(\log n) = s_{i-1} + O(\log(1/\varepsilon_{\mathsf{Ext}_i})),$$

and the recursive relation of the errors is $\varepsilon_i \le 6n^2\varepsilon_{i-1}^2$. Solving for the recursion, recalling $\varepsilon_0 = 1/w$, we get

$$\varepsilon_i \le \left(\frac{6n^2}{w}\right)^{2^i}.$$

Our choice of h guarantees a final error of $\varepsilon_h \le \varepsilon$.

By Theorem 13, the seed length

$$d_i = O\left(\log(w/\varepsilon_{\mathsf{Ext}_i})\right) = O(\log(1/\varepsilon_i)) = O\left(2^i \log w\right),$$

and so the recurrence simplifies to $s_i = s_{i-1} + O(2^i \log w)$, yielding

$$s_h = s_0 + O(2^h \log w) = s_0 + O(\log(1/\varepsilon)).$$

By Proposition 20,

$$s_0 = s(W) + O\left(\log\log\left(\frac{\log w}{\log n}\right)\cdot\log w\right),$$

and so the final seed length is

$$s_h = s(W) + O\left(\log\frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right)\cdot\log w\right).$$

We summarize this in the following theorem, which is the formal version of Theorem 1 whose proof follows by the above calculations, also noting that the constructed WPRGs remain error-dominated and explicit.

▶ **Theorem 21.** *Keeping the above notation, let W be an explicit error-dominated $(n, w, \Sigma, \Gamma_0, \varepsilon_0 = 1/n^5)$ WPRG with seed length $s(W)$. Then, V_h is an error-dominated $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG with seed length*

$$s(W) + O\left(\log\frac{1}{\varepsilon} + \log\log\left(\frac{\log w}{\log n}\right)\cdot\log w\right).$$

Finally, for our base-WPRG we'll choose Armoni's explicit PRG from Theorem 9.

▶ **Corollary 22.** *For any positive integers n, w such that $w > n$, any $\varepsilon > 0$, and any alphabet Σ, there exists an explicit $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG with seed length*

$$\log|\Sigma| + O\left(\frac{\log^2 n + \log n\cdot\log w}{\log\log w - \log\log n} + \log\log\left(\frac{\log w}{\log n}\right)\cdot\log w + \log\frac{1}{\varepsilon}\right).$$

and $\Gamma \le \max\left\{1, \frac{1}{\mathrm{poly}(n)\cdot\varepsilon}\right\}$. In particular, when $w = 2^{n^\alpha}$ for some constant $\alpha \in (0,1)$, the seed length above becomes

$$\log|\Sigma| + O\left((\log\log\log w)\cdot\log w + \log\frac{1}{\varepsilon}\right).$$

When w is only polynomial in n, we can perform the error reduction step of Section 4.4 directly, and for the base-PRG, we can use the simpler INW generator (see Theorem 8) and get the following corollary.

▶ **Corollary 23.** *For any positive integers n, w such that $w = n^{O(1)}$, any $\varepsilon > 0$, and any alphabet Σ, there exists an explicit $(n, w, \Sigma, \Gamma, \varepsilon)$ WPRG with seed length*

$$\log|\Sigma| + O\left(\log^2 n + \log\frac{1}{\varepsilon}\right)$$

and $\Gamma \le \max\left\{1, \frac{1}{\mathrm{poly}(n)\cdot\varepsilon}\right\}$.

References

1 Romas Aleliunas, Richard M. Karp, Richard J. Lipton, Laszlo Lovasz, and Charles Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *Annual Symposium on Foundations of Computer (FOCS)*, pages 218–223. IEEE, 1979.

2 Roy Armoni. On the derandomization of space-bounded computations. In *International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, volume 1518 of *LNCS*, pages 47–59. Springer, 1998. doi:10.1007/3-540-49543-6_5.

3 Allan Borodin, Stephen Cook, and Nicholas Pippenger. Parallel computation for well-endowed rings and space-bounded probabilistic machines. *Information and Control*, 58(1-3):113–136, 1983. doi:10.1016/S0019-9958(83)80060-6.

4 Mark Braverman, Gil Cohen, and Sumegha Garg. Pseudorandom pseudo-distributions with near-optimal error for read-once branching programs. *SIAM Journal on Computing*, 49(5):STOC18–242–STOC18–299, 2020. doi:10.1137/18M1197734.

5 Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. *SIAM Journal on Computing*, 43(3):973–986, 2014. doi:10.1137/120875673.

6 Eshan Chattopadhyay and Jyun-Jie Liao. Optimal error pseudodistributions for read-once branching programs. In *Computational Complexity Conference (CCC)*, pages 25:1–25:27. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CCC.2020.25.

7 Ben Chen, Gil Cohen, Dean Doron, Yuval Khaskelberg, and Amnon Ta-Shma. Improved error reduction for weighted PRGs. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2026. Manuscript.

8 Ben Chen and Amnon Ta-Shma. Simplifying Armoni's PRG. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 36:1–36:8. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.APPROX/RANDOM.2025.36.

9 Lijie Chen, William M. Hoza, Xin Lyu, Avishay Tal, and Hongxun Wu. Weighted pseudorandom generators via inverse analysis of random walks and shortcutting. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1224–1239. IEEE, 2023. doi:10.1109/FOCS57990.2023.00072.

10 Kuan Cheng and William M. Hoza. Hitting sets give two-sided derandomization of small space. *Theory of Computing*, 18(21):1–32, 2022. doi:10.4086/toc.2022.v018a021.

11 Kuan Cheng and Yichuan Wang. BPL $\subseteq$ L-AC^1. In *39th Computational Complexity Conference (CCC 2024)*, volume 300 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 32:1–32:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.CCC.2024.32.

12 Kuan Cheng and Ruiyang Wu. Weighted pseudorandom generators for read-once branching programs via weighted pseudorandom reductions. In *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3423–3458. SIAM, 2026. doi:10.1137/1.9781611978971.124.

13 Gil Cohen, Dean Doron, Tomer Manket, Edward Pyne, Yichuan Wang, and Tal Yankovitz. A study of error reduction polynomials. In *Electronic Colloquium on Computational Complexity (ECCC)*, 2025. Manuscript.

14 Gil Cohen, Dean Doron, Oren Renard, Ori Sberlo, and Amnon Ta-Shma. Error reduction for weighted PRGs against read once branching programs. In *Computational Complexity Conference (CCC)*, pages 22:1–22:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CCC.2021.22.

15 Gil Cohen, Dean Doron, Ori Sberlo, and Amnon Ta-Shma. Approximating iterated multiplication of stochastic matrices in small space. In *Annual Symposium on Theory of Computing (STOC)*, pages 35–45. ACM, 2023. doi:10.1145/3564246.3585181.

16 Anindya De. Pseudorandomness for permutation and regular branching programs. In *Computational Complexity Conference (CCC)*, pages 221–231. IEEE, 2011. doi:10.1109/CCC.2011.23.

17 Dean Doron, Pooya Hatami, and William M. Hoza. Near-optimal pseudorandom generators for constant-depth read-once formulas. In *34th Computational Complexity Conference (CCC 2019)*, pages 16:1–16:34. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.CCC.2019.16.

18 Dean Doron, Raghu Meka, Omer Reingold, Avishay Tal, and Salil Vadhan. Pseudorandom generators for read-once monotone branching programs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 58:1–58:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.APPROX/RANDOM.2021.58.

19 Dean Doron, Edward Pyne, and Roei Tell. Opening up the distinguisher: A hardness to randomness approach $\mathbf{BPL} = \mathbf{L}$ that uses properties of $\mathbf{BPL}$. In *Annual Symposium on Theory of Computing (STOC)*, pages 2039–2049. ACM, 2024.

20 Dean Doron and Roei Tell. Derandomization with Minimal Memory Footprint. In *38th Computational Complexity Conference (CCC 2023)*, volume 264 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.CCC.2023.11.

21 Oded Goldreich and Avi Wigderson. Tiny families of functions with random properties: A quality-size trade-off for hashing. *Random Struct. Algorithms*, 11(4):315–343, 1997. doi:10.1002/(SICI)1098-2418(199712)11:4\%3C315::AID-RSA3\%3E3.0.CO;2-1.

22 William M. Hoza. Better pseudodistributions and derandomization for space-bounded computation. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 28:1–28:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.APPROX/RANDOM.2021.28.

23 William M. Hoza, Edward Pyne, and Salil Vadhan. Pseudorandom generators for unbounded-width permutation branching programs. In *Innovations in Theoretical Computer Science Conference (ITCS)*, pages 7:1–7:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ITCS.2021.7.

24 William M. Hoza and David Zuckerman. Simple optimal hitting sets for small-success $\mathbf{RL}$. *SIAM Journal on Computing*, 49(4):811–820, 2020. doi:10.1137/19M1268707.

25 Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Annual Symposium on Theory of Computing (STOC)*, pages 356–364. ACM, 1994. doi:10.1145/195058.195190.

26 H. Jung. Relationships between probabilistic and deterministic tape complexity. In *Mathematical Foundations of Computer Science (MFCS)*, volume 118 of *LNCS*, pages 339–346. Springer, 1981. doi:10.1007/3-540-10856-4_101.

27 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. Revisiting norm estimation in data streams. *arXiv preprint arXiv:0811.3648*, 2008. arXiv:0811.3648.

28 Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002. doi:10.1137/S0097539700389652.

29 Michal Koucký, Prajakta Nimbhorkar, and Pavel Pudlák. Pseudorandom generators for group products. In *Annual Symposium on Theory of Computing (STOC)*, pages 263–272. ACM, 2011.

30 Daniel Kozlov. Simplifying the BCG PRPD for space bounded computation. Master's thesis, Tel-Aviv University, 2020. Available at https://www.cs.tau.ac.il/~amnon/Students/daniel.kozlov.pdf.

31 Raghu Meka, Omer Reingold, and Avishay Tal. Pseudorandom generators for width-3 branching programs. In *Annual Symposium on Theory of Computing (STOC)*, pages 626–637. ACM, 2019. doi:10.1145/3313276.3316319.

32 Jack Murtagh, Omer Reingold, Aaron Sidford, and Salil Vadhan. Derandomization beyond connectivity: Undirected laplacian systems in nearly logarithmic space. In *Annual Symposium on the Foundations of Computer Science (FOCS)*, pages 801–812. IEEE, 2017. doi:10.1109/FOCS.2017.79.

33 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992. doi:10.1007/BF01305237.

34 Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996. doi:10.1006/JCSS.1996.0004.

35 Aaron Putterman and Edward Pyne. Near-optimal derandomization of medium-width branching programs. In *Annual Symposium on Theory of Computing (STOC)*, pages 23–34. ACM, 2023.

36 Edward Pyne and Salil Vadhan. Pseudodistributions That Beat All Pseudorandom Generators. In *36th Computational Complexity Conference (CCC 2021)*, volume 200 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 33:1–33:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CCC.2021.33.

37 Ran Raz and Omer Reingold. On recycling the randomness of states in space bounded computation. In *31st Annual Symposium on Theory of Computing (STOC 1999)*, pages 159–168. ACM, 1999. doi:10.1145/301250.301294.

38 Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders. *Annals of Mathematics*, pages 157–187, 2002.

39 Michael E. Saks and Shiyu Zhou. $\mathbf{BP_HSPACE}(S) \subseteq \mathbf{DSPACE}(S^{2/3})$. *Journal of Computer and System Sciences*, 58(2):376–403, 1999.

40 Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970. doi:10.1016/S0022-0000(70)80006-X.

Output-Sparse Matrix Multiplication Using Compressed Sensing

Huck Bennett
University of Colorado Boulder, CO, USA

Karthik Gajulapalli
Georgetown University, Washington, DC, USA

Alexander Golovnev
Georgetown University, Washington, DC, USA

Evelyn Warton
Oregon State University, Corvallis, OR, USA

Abstract

We give two algorithms for *output-sparse matrix multiplication* (OSMM), the problem of multiplying two $n \times n$ matrices A, B when their product AB is promised to have at most $O(n^\delta)$ many non-zero entries for a given value $\delta \in [0, 2]$. We then show how to speed up these algorithms in the *fully sparse matrix multiplication* (FSMM) setting, where the input matrices A, B are themselves sparse. All of our algorithms work over arbitrary rings.

Our first, deterministic algorithm for OSMM works via a two-pass reduction to compressed sensing. It runs in roughly $n^{\omega(\delta/2,1,1)}$ time, where $\omega(\cdot,\cdot,\cdot)$ is the rectangular matrix multiplication exponent. This substantially improves on prior deterministic algorithms for output-sparse matrix multiplication.

Our second, randomized algorithm for OSMM works via a reduction to compressed sensing and a variant of matrix multiplication verification, and runs in roughly $n^{\omega(\delta-1,1,1)}$ time. This algorithm and its extension to the fully sparse setting have running times that match those of the (randomized) algorithms for OSMM and FSMM, respectively, in recent work of Abboud, Bringmann, Fischer, and Künnemann (SODA, 2024). Our algorithm is quite simple when taking a suitable compressed sensing scheme as a black box.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Pseudorandomness and derandomization; Theory of computation → Expander graphs and randomness extractors; Theory of computation → Error-correcting codes

Keywords and phrases Matrix Multiplication, Sparse Matrices, Compressed Sensing

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.40

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2508.10250` [6]

Funding *Huck Bennett*: Supported by NSF Grant CCF-2432132.
Karthik Gajulapalli: Supported by NSF grant CCF-2338730.
Alexander Golovnev: Supported by NSF grant CCF-2338730.

Acknowledgements HB and AG would like to thank Divesh Aggarwal, the National University of Singapore (NUS), and the Centre for Quantum Technologies (CQT). Some of this work was completed while visiting them. The authors would also like to thank Thatchaphol Saranurak for suggesting the use of the expander construction in [17], and Noah Stephens-Davidowitz for helpful comments. Finally, the authors would like to thank anonymous reviewers for providing helpful references and comments (in particular, for noting the connection between [1, Lemma 3.1] and our work).



© Huck Bennett, Karthik Gajulapalli, Alexander Golovnev, and Evelyn Warton;
licensed under Creative Commons License CC-BY 4.0



1 Introduction

The Matrix Multiplication Problem (MM) is to compute the product AB of two $n \times n$ matrices A, B given as input. It is one of the most fundamental problems in computer science. The running time $O(n^\omega)$ of the fastest algorithm for MM is governed by the *matrix multiplication exponent* $\omega \leq 2.372$ (see [2] for the current best bound), and it is a major open question whether $\omega = 2$.

While one line of work has sought to improve ω, another has tried to find faster algorithms for important special cases of MM. In this work, we give new algorithms for the *Output-Sparse Matrix Multiplication Problem* (OSMM), where the product AB of the input matrices A and B is promised to be *sparse*. Specifically, we consider MM when AB is promised to have $O(n^\delta)$ many non-zero entries for a parameter $\delta \in [0, 2]$. We denote this problem for matrices over a ring R by δ-OSMM$_R$. We also discuss how to extend our algorithms to the *fully sparse* setting, in which the input matrices A and B are themselves sparse. This setting was recently studied in depth in seminal work of Abboud, Bringmann, Fischer, and Künnemann [1].

1.1 Our Work

Our main results are a deterministic algorithm and a randomized algorithm for OSMM over arbitrary rings R including, in particular, finite fields, the integers, and the real numbers. The algorithms perform a similar number of ring operations regardless of R.

Define the *rectangular matrix multiplication exponent* $\omega(\alpha, \beta, \gamma)$ to be the infimum over values $\omega' \geq 0$ such that there is an algorithm for multiplying an $n^\alpha \times n^\beta$ matrix and an $n^\beta \times n^\gamma$ matrix in $O(n^{\omega'})$ time. Define the *dual matrix multiplication exponent* $\omega^\perp$ to be the supremum over values $\omega' \geq 0$ such that $\omega(1, 1, \omega') = 2$.[1]

The theorems corresponding to our main two OSMM algorithms are as follows. Our first algorithm dramatically improves on the state of the art for deterministic algorithms; see the comparison with prior work in Figure 1 and Section 1.2.

▶ **Theorem 1** (Deterministic OSMM). *Let $\delta \in [0, 2]$, let R be a ring, and let $\varepsilon > 0$ be a constant. There is a deterministic algorithm for solving δ-OSMM$_R$ on $n \times n$ matrices that performs $O(n^{\omega(\delta/2,1,1)+\varepsilon})$ operations over R.*

Our second, randomized algorithm, presented in Theorem 2, has running time that matches an algorithm from [1] using a reduction to a variant of the Matrix Multiplication Verification Problem (MMV). See Section 1.2 for a comparison of our Theorem 2 and the algorithm in [1]. We show that the running times of these algorithms are optimal in Section 4.3.

▶ **Theorem 2** (Randomized OSMM). *Let $\delta \in [0, 2]$, let R be a ring, and let $\varepsilon > 0$ be a constant. There is a randomized algorithm for solving δ-OSMM$_R$ on $n \times n$ matrices that performs $O(n^{\beta+\varepsilon})$ operations over R, where*

$$\beta := \begin{cases} 2 & \text{if } \delta \leq 1 + \omega^\perp \ , \\ \omega(\delta - 1, 1, 1) & \text{otherwise} \ . \end{cases}$$

[1] The dual matrix multiplication exponent is often denoted by α in the literature.

Since $\omega^{\perp} \geq 0.321$ [35], the algorithms in Theorems 1 and 2 run in essentially optimal n^2 time for all $\delta \leq 0.642$ and $\delta \leq 1.321$, respectively. Furthermore, both algorithms run in $O(n^{\omega-\varepsilon})$ time for some $\varepsilon = \varepsilon(\delta) > 0$ for every $\delta < 2$.[2] That is, both algorithms are faster than the fastest known algorithms for matrix multiplication as long as AB is promised to be even, say, $n^{1.99}$-sparse. We also extend our algorithms to the *fully sparse* setting where the input matrices A, B are themselves sparse (in addition to having the promise that their product AB is sparse), and get faster algorithms in this case; see Section 4 and Theorems 18 and 19. Although we choose to highlight our somewhat simpler OSMM algorithms, they actually follow as corollaries from our algorithms for the fully sparse case.

We note that our extension of Theorem 2 to the fully sparse setting (given in Theorem 19) is equivalent to [1, Lemma 3.1], the main technical lemma in [1] for fully sparse matrix multiplication on general rings. However, our corresponding algorithm uses different techniques and is arguably simpler. In Proposition 20 we also show that our randomized algorithms in Theorems 2 and 19 (and therefore also [1, Lemma 3.1]) are *optimal* via a reduction from rectangular matrix multiplication.

We also provide an exposition of a nearly optimal, deterministic compressed sensing that works over any ring in an appendix of the full version of this paper [6]. The scheme instantiates the compressed sensing scheme from [7] with the expander construction from [17]. Although likely known as folklore, this scheme does not seem to have been analyzed in detail before.

1.2 Applications and Comparison with Prior Work

Input- and output-sparse matrix multiplication have a number of applications including answering certain database queries [4, 34, 10], computing transitive closures in sparse graphs [34, 8], multi-source breadth-first search [37, 16, 13], Markov clustering [33], and error correction of matrix products [14, 23, 32].[3]

A long line of work has studied both input- and output-sparse matrix multiplication [18, 36, 20, 28, 4, 30, 24, 21, 34, 23, 32, 10, 1]. Our deterministic algorithm in Theorem 1 substantially improves on prior deterministic δ-OSMM algorithms for most values of δ, and is never slower than them. See Figure 1. As noted above, our algorithm runs in essentially optimal quadratic time for $\delta \leq 0.642$, and for every constant $\delta < 2$ it runs in $O(n^{\omega-\varepsilon})$ time for some $\varepsilon = \varepsilon(\delta) > 0$.

The main prior works on deterministic OSMM are Kutzkov [24], which gives a $O(n^2 + n^{2\delta+1})$-time algorithm for OSMM over the reals, and Künnemann [23], which gives a $O(n^{2+\delta/2} + n^{2\delta})$-time algorithm for OSMM over the integers. Their algorithms only run in $O(n^{\omega-\varepsilon})$ time for $\delta < (\omega-1)/2 \approx 0.686$ and $\delta < 2(\omega-2) \approx 0.744$, respectively. (One can also show analytically that Theorem 1 is at least as fast as [24, 23] for all δ.[4]) Moreover, if one is interested in only using "combinatorial algorithms" for matrix multiplication (i.e., assuming that $m \times n \times p$ rectangular MM takes $O(mnp)$ time), our deterministic algorithm runs in $\widetilde{O}(n^{2+\delta/2})$ time. So, our algorithm is strictly faster than the algorithms in [24, 23] for $\delta > 4/3$.

[2] Note that $\delta - 1 \leq \delta/2$ for $\delta \in [0, 2]$ so the randomized algorithm in Theorem 2 is always at least as fast as the deterministic algorithm in Theorem 1.

[3] We note that error correction of matrix products is equivalent to OSMM.

[4] By Theorem 7 we have that $\omega(1, 1, \delta/2) \leq 1.825 + 0.548\delta/2$ from which it follows that $\omega(1, 1, \delta/2) \leq 2\delta + 1$ for all $\delta \geq 0.5$. So, our algorithm is always at least as fast as the algorithm in [24]. Furthermore, by Proposition 6, Item 1, $\omega(\delta/2, 1, 1) \leq 2 + \delta/2$ and so our algorithm is always at least as fast as the algorithm in [23].

Our randomized OSMM algorithm in Theorem 2 and the breakthrough randomized algorithm in [1] both run in roughly $n^{\omega(\delta-1,1,1)}$ time. They work in a similar way at a very high level, however there are notable differences. Perhaps more starkly, unlike [1], our randomized algorithm works by reducing OSMM to a variant of the Matrix Multiplication Verification (MMV) problem. Moreover, solving this variant of MMV is the algorithm's *only* use of randomness.

The algorithms in Theorem 2 and [1] both work by (implicitly) partitioning the output matrix AB into blocks of rows or columns with similar sparsity, "densifying" corresponding blocks in the input matrices, multiplying these dense blocks, and then recovering the (sparse) output matrix. To do this, [1] uses hash functions for the densification and a method for computing a small overapproximation of the support of the output matrix. On the other hand, our randomized OSMM algorithm, given in Algorithm 2, works directly with the primitives of a compressed sensing scheme, and it is simple to describe and analyze when these are taken as a black box. In particular, our algorithm performs densification by multiplying by a measurement matrix, and recovers (parts of) the output matrix via sparse recovery. (One can also interpret the hashing-based densification method in [1] as multiplying by a certain (random) measurement matrix. Our algorithm and analysis abstract this out.) See Section 1.3 for a detailed technical overview of our algorithms.

We also note a connection between our algorithm for OSMM and an algorithm from [5] for the All Zeroes Problem, the problem of checking whether $AB = 0$ for two $n \times n$ input matrices A, B. (We note that [23] showed that the All Zeroes Problem and the better-known MMV problem are essentially equivalent under a nearly sparsity-preserving reduction.) Specifically, Theorem 1 achieves the same running time of roughly $n^{\omega(\delta/2,1,1)}$ for δ-OSMM that a deterministic algorithm in [5] achieved for the All Zeroes Problem with the same promise that the input matrices A, B satisfy $\|AB\|_0 \leq O(n^\delta)$. The All Zeroes Problem with the guarantee $\|AB\|_0 \leq O(n^\delta)$ reduces to δ-OSMM (this is a trivial decision-to-search reduction), and so one can view Theorem 1 as an upgrade of the deterministic algorithm in [5]. We also emphasize again that Theorems 1 and 2 work over *any* ring R whereas the algorithms from each of [24, 23, 5] only work on specific domains.

The sole deterministic OSMM algorithm that is faster than ours is an algorithm that (only) works for multiplication over nonnegative integers in [1, Lemma 3.11].[5] Their algorithm's running time is stated for the fully sparse setting, but it runs in roughly $O(n^{\omega(\delta-1,1,1)})$ time in the output sparse setting. This matches the running time of randomized algorithms in their work and ours for OSMM over general rings.

1.3 Technical Overview

We next give an overview of our algorithms, which use simple primitives: fast rectangular matrix multiplication, a compressed sensing scheme with good parameters, and an algorithm for a variant of matrix multiplication verification. Define $\|\boldsymbol{v}\|_0$ (respectively, $\|M\|_0$) to be the number of non-zero entries in (i.e., Hamming weight of) a vector $\boldsymbol{v}$ (respectively, matrix M). We call a vector $\boldsymbol{v}$ (respectively, matrix M) *t-sparse* if $\|\boldsymbol{v}\|_0 \leq t$ (respectively, if $\|M\|_0 \leq t$).

[5] We note that algorithms for matrix multiplication over nonnegative integers imply algorithms for Boolean matrix multiplication as a special case. Recall that the (i, j)th entry of the Boolean product of matrices $A, B \in \{0, 1\}^{n\times n}$ is defined as $\vee_{k=1}^{n}(A_{i,k} \wedge B_{k,j})$. To compute this product it suffices to compute the standard matrix product AB over the integers and replace all positive entries of the output with ones.

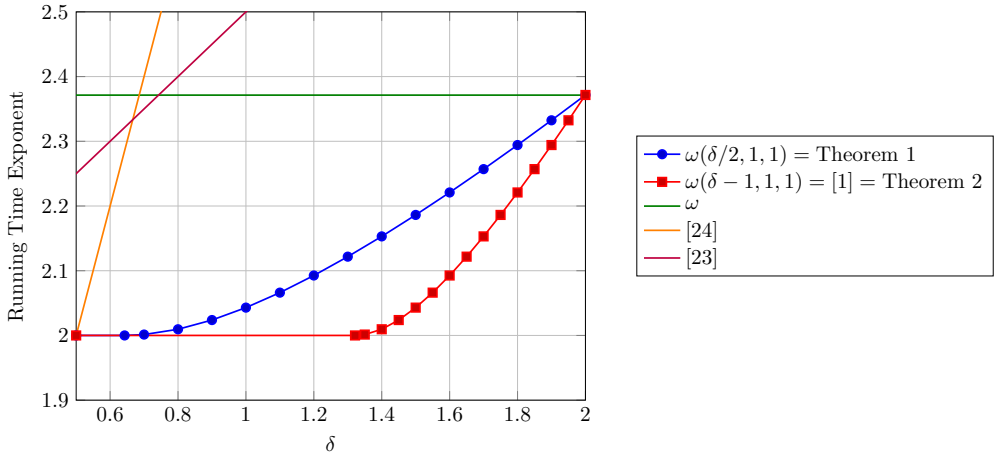


Figure 1 A plot of the running time exponents of several algorithms for δ-OSMM, with arbitrarily small polynomial factors suppressed. The blue curve gives the running time of our deterministic algorithm from Theorem 1, which runs in $n^{\omega(\delta/2,1,1)}$ time. The red curve gives the running time of the randomized algorithm from [1] and our randomized algorithm from Theorem 2. These algorithms both run in roughly $n^{\omega(\delta-1,1,1)}$ time. The points on the curves come from upper bounds on rectangular MM exponents in [35], and the line segments connecting them are justified by the convexity of $\omega(\cdot, 1, 1)$. We also plot the running time exponents of the $O(n^{\max\{2,2\delta+1\}})$-time algorithm of Kutzkov [24] and the $O(n^{\max\{2+\delta/2,2\delta\}})$-time algorithm of Künnemann [23] – which are the main prior works giving *deterministic* algorithms for OSMM – as well as the MM exponent ω.

1.3.1 Output-column-sparse MM using compressed sensing

In the *compressed sensing* problem with parameters $n, m, t \in \mathbb{Z}^+$ with $1 \le t \le n$, the goal is to compute an $m \times n$ matrix $H = H(m, n, t)$ called a *measurement matrix*, and to design an efficient *sparse recovery* algorithm $\mathcal{R}$ (depending on H) such that for every t-sparse vector $\boldsymbol{x}$ of length n, $\mathcal{R}(H\boldsymbol{x}) = \boldsymbol{x}$. That is, the goal is for $\mathcal{R}$ to be able to recover any t-sparse vector $\boldsymbol{x}$ from $H\boldsymbol{x}$.

Several previous works including [20] use compressed sensing to solve MM in the case where every column $\boldsymbol{c}_i$ of the output matrix $C = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n) := AB$ is promised to be t-sparse. The technique works by computing HAB parenthesized as $(HA)B$ using fast rectangular matrix multiplication, and then recovering the columns $\boldsymbol{c}_i$ of $C = AB$ from HC using $\mathcal{R}$ as $\boldsymbol{c}_i = \mathcal{R}(H\boldsymbol{c}_i)$ for $i = 1, \ldots, n$. If computing H takes $T_{\mathcal{M}}(n)$ time, $\mathcal{R}$ takes $T_{\mathcal{R}}(n)$ time, and $m = O(n^\beta)$ for some $\beta \in [0, 1]$, then for any constant $\varepsilon > 0$ the overall running time of this algorithm is

$$O(T_{\mathcal{M}}(n) + n^{\omega(\beta,1,1)+\varepsilon} + n \cdot T_{\mathcal{R}}(n)) \ . \tag{1}$$

We note that [20] uses the compressed sensing scheme of [19]. The compressed sensing scheme there has good parameters and has a fast sparse recovery algorithm $\mathcal{R}$. However, while [19] uses a polynomial-time construction of expanders, it is unclear that its construction runs in better-than-n^ω time. (We achieve $T_{\mathcal{M}}(n) = \widetilde{O}(n)$ time in Theorem 13.)

1.3.2 Reducing OSMM to compressed sensing

As our first contribution, we show how to replace the assumption in the above compressed-sensing-based algorithm for MM that every column of $C = AB$ is t-sparse with the "global" assumption that C is t^2-sparse. If $t^2 = O(n^\delta)$, this is exactly the assumption in δ-OSMM.

To do this, we use a two-pass reduction, which roughly speaking first computes a matrix C' that agrees with C on sparse columns and then computes all rows of $C - C'$. This reduction appears formally as Algorithm 1.

Assume without loss of generality that $\mathcal{R}$ always outputs *some* vector of length n even when its input is not t-sparse. More specifically, assume that $\boldsymbol{c}'_i := \mathcal{R}(H\boldsymbol{c}_i)$ is always a vector of length n with the guarantee that $\boldsymbol{c}'_i = \boldsymbol{c}_i$ if $\boldsymbol{c}_i$ is t-sparse. Let $C' := (\boldsymbol{c}'_1, \ldots, \boldsymbol{c}'_n)$. The key observation is that if $C = AB$ is t^2-sparse, then *every* row of $C - C'$ must be t-sparse.[6] (Similar observations were previously used, e.g., in the proof of [14, Theorem 4.3] and in [5].) To see this, note that if C is t^2-sparse, then it has fewer than t columns that are *not* t-sparse, and that these are the only (potentially) non-zero columns in $C - C'$. We can then recover the entries of $(C - C')^T$ (and therefore also C, since we know C') by computing $H(C - C')^T$ as $(HB^T)A^T - H(C')^T$ and then running $\mathcal{R}$ on each column of the result.

1.3.3 Reducing OSMM to compressed sensing and a variant of MMV

Our second reduction is from OSMM to compressed sensing and a variant of the matrix multiplication verification problem (MMV) that we call Column-Wise-MMV. The goal of Column-Wise-MMV is, given three matrices A, B, and C' as input, to identify all of the indices $i \in [n]$ such that $\boldsymbol{c}_i \neq \boldsymbol{c}'_i$, where $C = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n) := AB$ and $C' = (\boldsymbol{c}'_1, \ldots, \boldsymbol{c}'_n)$. This reduction appears formally as Algorithm 2.

The main idea of our second reduction is to sequentially "guess and check" all columns of C of sparsity at most 2^i for $i = 0, \ldots, \lceil \log n \rceil$. Namely, in iteration i we use a compressed sensing scheme as above to compute a matrix C' that agrees with all columns of C of sparsity at most $t = 2^i$. (As before, we assume that $\mathcal{R}$ always outputs some vector of length n regardless of its input.) We then use our Column-Wise-MMV oracle to identify which of the columns of C were computed correctly in C'. We use this information to update the corresponding columns of our output matrix and to remove the corresponding columns from B. We emphasize that this reduction itself is deterministic, and the only use of randomness in the algorithm summarized in Theorem 2 comes from solving Column-Wise-MMV.

Correctness follows simply by analyzing the last iteration (in which $i = \lceil \log n \rceil$) since each column of C has sparsity at most $n \leq 2^{\lceil \log n \rceil}$. In fact, the reduction does not require an upper bound on the sparsity of C as input and works regardless of this sparsity – it is a reduction from MM, not just OSMM. However, the sparser C is the faster the reduction is. Let $J_{i+1} \subseteq [n]$ be the set of indices of columns of C with more than 2^i non-zero entries.[7]

Assume that it is possible to multiply the $m \times n$ measurement matrix H by an arbitrary $n \times n$ matrix A in $O(n^{2+\varepsilon})$ time for any constant $\varepsilon > 0$. When we instantiate the reduction with the concrete compressed sensing scheme described in Theorem 13 this will be the case since our measurement matrices H will be sparse. Then the bottleneck in each iteration i is computing the product of HA (an $m \times n$ matrix) and B (an $n \times |J_i|$ matrix). Let $\alpha := i/\log n$. We know that $|J_{i+1}| \leq O(n^\delta/2^i) = O(n^{\delta-\alpha})$, and if $m \approx t = 2^i = n^\alpha$ then it is possible to compute the product of HA and B in roughly $n^{\omega(\alpha,1,\delta-\alpha)}$ time. By Corollary 8,

[6] Note that it is *not* the case that every non-zero entry of C must be contained in a t-sparse row or column. For example, if $C = (\boldsymbol{e}_1 \cdot \mathbf{1}^T) + (\mathbf{1} \cdot \boldsymbol{e}_1) \in \mathbb{Z}^{n \times n}$ then $C_{1,1} = 2$ and both the first column and first row of C have n non-zero entries. (Here $\boldsymbol{e}_1$ and $\mathbf{1}$ denote the first standard normal basis vector and the all-ones vector, respectively.) Yet, C is $O(n)$-sparse.

[7] In Algorithm 2, J_{i+1} can technically be a subset of these columns if the sparse recovery algorithm happens to succeed (i.e., when $\mathcal{R}(H\boldsymbol{x}) = \boldsymbol{x}$) at an earlier iteration $j < i$ even when $\boldsymbol{x}$ is not 2^j-sparse. But, $|J_{i+1}|$ being smaller only makes the algorithm run faster.

$\omega(\alpha, 1, \delta - \alpha) \leq \omega(\delta - 1, 1, 1)$ for any α, and so the claimed running time of $O(n^{\omega(\delta-1,1,1)+\varepsilon})$ follows since we only need to compute $O(\log n)$ such matrix products and the multiplicative $\log n$ factor gets absorbed by the ε in the exponent.

1.3.4 From Output-Sensitive MM to Fully Sparse MM

In Section 4, we extend our algorithms to the *fully sparse* case where we consider the sparsity of the $n \times n$ input matrices A, B in addition to the sparsity of AB. I.e., we are promised not only that $\|AB\|_0 \leq O(n^{\delta_{\text{out}}})$ for some $\delta_{\text{out}} \in [0, 2]$, but also that $\|A\|_0, \|B\|_0 \leq O(n^{\delta_{\text{in}}})$ for $\delta_{\text{in}} \in [0, 2]$.[8] For $\alpha, \beta, \gamma \in [0, 1]$, define $\omega_{\delta_{\text{in}}}(\alpha, \beta, \gamma)$ to be the infimum over $\omega' \geq 0$ such that it is possible to compute the product of an $n^\alpha \times n^\beta$ matrix A and an $n^\beta \times n^\gamma$ matrix B in $O(n^{\omega'})$ time when $\|A\|_0, \|B\|_0 \leq O(n^{\delta_{\text{in}}})$. We note that there are algorithms showing that $\omega_{\delta_{\text{in}}}(\alpha, \beta, \gamma)$ is smaller than $\omega(\alpha, \beta, \gamma)$ for sufficiently small values of δ_{in} [36, 22].

Theorem 18 (respectively, Theorem 19) extends our deterministic (respectively, randomized) algorithm in Theorem 1 (respectively, Theorem 2) to the fully sparse setting. Theorem 18 achieves a running time of roughly $O(n^{\omega_{\delta_{\text{in}}}(\delta_{\text{out}}/2,1,1)} + n^{\delta_{\text{out}}/2+1})$, which is equal to the $O(n^{\omega(\delta_{\text{out}}/2,1,1)})$ running time of Theorem 1 when $\delta_{\text{in}} = 2$, i.e., when there is no promise on the sparsity of A, B as in δ_{out}-OSMM.

Theorem 19 achieves a running time of roughly $O(n^{\delta_{\text{in}}} + n^\beta)$, for

$$\beta := \max_{\eta \in [0, \min\{\delta_{\text{out}}, 1\}]} \omega_{\delta_{\text{in}}}(\eta, 1, \delta_{\text{out}} - \eta) \ .$$

Although this expression looks intricate, it appears in [1, Lemma 3.1] (to which it is equivalent) and arises naturally in our FSMM proof of optimality; see Proposition 20. (For $\delta_{\text{out}} \geq 1$ and $\delta_{\text{in}} = 2$, we also get that $\omega_{\delta_{\text{in}}}(\eta, 1, \delta_{\text{out}} - \eta) \leq \omega(\delta_{\text{out}} - 1, 1, 1)$ by Corollary 8, and moreover $\omega(\delta_{\text{out}} - 1, 1, 1) \geq 2 \geq \delta_{\text{in}}$. So, we achieve the running time of Theorem 2.)

Extending our algorithms to the fully sparse setting essentially only uses basic facts about input-sparse matrix multiplication and properties of our compressed sensing scheme. Specifically, it uses the fact that our measurement matrix H has very sparse columns, and that on input $H\boldsymbol{x}$ for a t-sparse vector $\boldsymbol{x}$ of length n, $\mathcal{R}$ runs in time roughly t rather than time roughly n. In particular, we use that such a matrix H admits fast multiplication by any matrix A, and if A is sparse then HA is still nearly as sparse. (To achieve all of this, we use sparse representations of matrices and vectors.)

1.3.5 Instantiating our reductions

To instantiate our reductions and turn them into the algorithms summarized by Theorems 1 and 2, we need to give an explicit compressed sensing scheme and an algorithm for Column-Wise-MMV.

We use a compressed sensing scheme due to Berinde, Gilbert, Indyk, Karloff and Strauss [7], which requires certain expander graphs. To make the scheme of [7] deterministic and obtain good parameters, we instantiate it with the expander graphs later studied in Guruswami, Umans, and Vadhan [17]. In particular, our measurement matrices H correspond to the adjacency matrices of the bipartite expanders constructed in [17] from Parvaresh-Vardy codes. They are sparse and binary. The main theorem summarizing this scheme is Theorem 13.

[8] Without loss of generality we can take $\delta_{\text{out}} \leq 2\delta_{\text{in}}$.

For completeness, we present and analyze the expander construction in [17] and the sparse recovery algorithm in [7] in an appendix of the full version of this paper [6]. We show that they are both very efficient. Additionally, we observe that the sparse recovery algorithm in [7] works not just over the real numbers (as it was originally presented), but over arbitrary rings.

We emphasize the subtlety of identifying an appropriate compressed sensing scheme for our purposes. We need algorithms that compute the measurement matrix and perform sparse recovery not just in (arbitrary) polynomial time, but in nearly quadratic and nearly linear time, respectively. Moreover, we need these algorithms to be deterministic – many works on compressed sensing sample a random measurement matrix H (see, e.g., [9, 15]). Finally, for Theorem 2 we need multiplication by H to be very efficient.

The algorithm for Column-Wise-MMV that we use (stated formally in Lemma 16) is a modified version of Freivalds's algorithm for "plain" MMV [12]. This algorithm is the only use of randomness in our our randomized OSMM algorithm, and so if it could be derandomized efficiently then we would get a *deterministic* algorithm for δ-OSMM running in roughly $n^{\omega(\delta-1,1,1)}$ time. (Similarly, this algorithm is the only use of randomness in our randomized FSMM algorithm.)

1.3.6 An alternative randomized algorithm

We note that it is possible to implement a variant of our randomized algorithm that achieves a similar running time to Theorem 2 if it is possible to approximate the number of non-zero entries in each column of $C = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n) = AB$ to within a constant factor efficiently.

In fact, we can compute estimates $\tilde{c}_i$ of the sparsity of each column $\boldsymbol{c}_i$ satisfying $1/2 \cdot \|\boldsymbol{c}_i\|_0 < \tilde{c}_i < 2 \cdot \|\boldsymbol{c}_i\|_0$ with high probability using a randomized algorithm from [31]. We can then use these estimates to multiply A and B efficiently. We first partition the columns of B (and therefore AB) into blocks B_j, where B_j consists of all columns $\boldsymbol{b}_i$ such that $2^{j-1} \leq \tilde{c}_i < 2^j$ for $j = 1, \ldots, r$, where $r \approx \log_2 n$. We then compute AB_j for each j using the algorithm for output-column-sparse matrix multiplication sketched earlier using the assumption (which holds with high probability) that each column of AB_j is 2^{j+1}-sparse.

We note that the column-sparsity-estimation-based algorithm sketched here uses randomness in a different way from Theorem 2; it uses randomness to estimate column sparsities and does not need to solve Column-Wise-MMV. However, we believe that the reduction from OSMM to Column-Wise-MMV that we use to prove Theorem 2 – which can be viewed as a search-to-decision reduction of sorts – is elegant and more amenable to derandomization. In particular, a strong resolution to the famous question of derandomizing Freivalds's algorithm would derandomize Theorem 2 as well.

2 Preliminaries

By $\mathbb{Z}^+$ we denote the set of positive integers. For $n \in \mathbb{Z}^+$, $[n]$ denotes the set $\{1, \ldots, n\}$. Let R be a ring, $\boldsymbol{v} \in R^n$ be a vector, and $M \in R^{m \times n}$ be a matrix. Then by $\|\boldsymbol{v}\|_0$ and $\|M\|_0$ we denote the number of non-zero entries in $\boldsymbol{v}$ and M, respectively. We say $\boldsymbol{v}$ (respectively, matrix M) is *t-sparse* if $\|\boldsymbol{v}\|_0 \leq t$ (respectively, if $\|M\|_0 \leq t$). We say that M is *t-column-sparse* if each of its columns is t-sparse. For an $n \in \mathbb{Z}^+$, $\mathbf{0}_n$ and $\mathbf{1}_n$ denote n-dimensional vectors with all zeros and all ones, respectively.

We use $\log(\cdot)$ to denote the logarithm base 2, i.e., $\log(2^n) = n$. We use the notation $\widetilde{O}(\cdot)$ to suppress polylogarithmic factors in the argument.

We state the running time bounds of our algorithms in terms of operations over a ring R, but these bounds immediately imply algorithms with similar running times (in the usual sense) over many standard domains. Over the integers with entries in $[-M, M]$ each operation requires at most $\mathrm{poly}(\log M, \log n)$ time, and over finite fields $\mathbb{F}_q$ each operations requires at most $\mathrm{poly}(\log q, \log n)$ time. Assigning unit cost to operations on real numbers is standard in the Real RAM model.

2.1 Matrix Multiplication

We first define the main two variants of matrix multiplication that we study.

▶ **Definition 3** (Output and Fully Sparse Matrix Multiplication). *Let R be a ring, and let $\delta_{in}, \delta_{out} \in [0, 2]$. We define the $(\delta_{in}, \delta_{out})$-*Fully Sparse Matrix Multiplication Problem *over R ($(\delta_{in}, \delta_{out})$-FSMM$_R$) as follows. The input consists of $A, B \in R^{n\times n}$ for some $n \in \mathbb{Z}^+$ with $\|A\|_0, \|B\|_0 \le O(n^{\delta_{in}})$ satisfying the promise that $\|AB\|_0 \le O(n^{\delta_{out}})$. The goal is to compute AB.*

*We additionally define the δ_{out}-*Output-Sparse Matrix Multiplication Problem *over R (δ_{out}-OSMM$_R$) as $(2, \delta_{out})$-FSMM$_R$.*

When both $\delta_{\mathrm{in}} = 2$ and $\delta_{\mathrm{out}} = 2$ this corresponds to the problem of matrix multiplication in its general form. Furthermore, note that since the product of two t-sparse matrices is t^2-sparse, we may without loss of generality assume that $\delta_{\mathrm{out}} \le 2\delta_{\mathrm{in}}$.

We next define rectangular matrix multiplication exponents, input-sparse rectangular matrix multiplication exponents, and the dual matrix multiplication exponent.

▶ **Definition 4** (Rectangular and Input-Sparse Rectangular Multiplication Exponents). *For constants $\alpha, \beta, \gamma \ge 0$, the* rectangular matrix multiplication exponent $\omega(\alpha, \beta, \gamma)$ *is the infimum over all $\omega' \ge 0$ such that the product of $A \in \mathbb{Z}^{n^\alpha \times n^\beta}$ and $B \in \mathbb{Z}^{n^\beta \times n^\gamma}$ can be computed using $O(n^{\omega'})$ arithmetic operations.*

We extend this notation and use $\omega_{\delta_{in}}(\alpha, \beta, \gamma)$ to denote the case when both A and B are promised to be $O(n^{\delta_{in}})$-sparse for $0 \le \delta_{in} \le \max\{\alpha + \beta, \beta + \gamma\}$.

We note that the *square matrix multiplication exponent* ω is simply $\omega := \omega(1, 1, 1)$, and that $\omega_{\delta_{\mathrm{in}}}(\alpha, \beta, \gamma) \le \omega_{\max\{\alpha+\beta, \beta+\gamma\}}(\alpha, \beta, \gamma) = \omega(\alpha, \beta, \gamma)$ for all $\alpha, \beta, \gamma \ge 0$.

▶ **Definition 5** (Dual Matrix Multiplication Exponent). *The* dual matrix multiplication exponent $\omega^\perp$ *is defined as*

$$\omega^\perp := \sup\{\omega' \ge 0 : \omega(1, 1, \omega') = 2\} .$$

A beautiful line of work [27, 3, 11, 26, 35, 2] has established strong bounds on multiplication exponents, in particular showing that

$$\omega < 2.371339 \quad [2] ,$$
$$\omega^\perp \ge 0.321334 \quad [35] .$$

2.2 Properties of Matrix Multiplication Exponents

Next we list several properties of rectangular matrix multiplication exponents $\omega(\cdot, \cdot, \cdot)$.

▶ **Proposition 6** ([29]). *Let $\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3, \lambda \ge 0$. The following hold:*

1. $\max(\alpha_1 + \alpha_2, \alpha_1 + \alpha_3, \alpha_2 + \alpha_3) \le \omega(\alpha_1, \alpha_2, \alpha_3) \le \alpha_1 + \alpha_2 + \alpha_3$.
2. $\omega(\lambda\alpha_1, \lambda\alpha_2, \lambda\alpha_3) = \lambda \cdot \omega(\alpha_1, \alpha_2, \alpha_3)$.

3. $\omega(\alpha_1+\beta_1,\alpha_2+\beta_2,\alpha_3+\beta_3) \le \omega(\alpha_1,\alpha_2,\alpha_3)+\omega(\beta_1,\beta_2,\beta_3)$.
4. *For any permutation* $\pi:[3]\to[3]$, $\omega(\alpha_1,\alpha_2,\alpha_3)=\omega(\alpha_{\pi(1)},\alpha_{\pi(2)},\alpha_{\pi(3)})$.
5. $\omega(\alpha,\beta,\gamma)$ *is continuous and non-decreasing in its arguments for* $\alpha,\beta,\gamma \ge 0$.

We will make use of the following upper bound on $\omega(\alpha,1,1)$.

▶ **Theorem 7** ([29, 25]). *Let* $\alpha\in[0,1]$. *Then*

$$\omega(\alpha,1,1) \le \begin{cases} 2 & \text{if } 0\le\alpha\le\omega^\perp\ , \\ 2+\dfrac{\omega-2}{1-\omega^\perp}\cdot(\alpha-\omega^\perp) \le 1.825+0.548\alpha & \text{if } \omega^\perp<\alpha\le 1\ . \end{cases}$$

We next give an upper bound on rectangular matrix multiplication exponents $\omega(1,\alpha,\beta)$ with three potentially distinct arguments.

▶ **Corollary 8.** *Let* $\alpha,\beta,\gamma\in[0,1]$ *be such that* $\beta+\gamma\ge 1$. *Then* $\omega(\alpha,\beta,\gamma)\le\omega(\alpha,1,\beta+\gamma-1)$.

Proof. If $\beta=\gamma=1$, the statement holds trivially. Otherwise we can assume that $\beta+\gamma<2$. Let $\lambda=(1-\gamma)/(2-\beta-\gamma)\ge 0$. Then we have that

$$\begin{aligned}\omega(\alpha,\beta,\gamma) &= \omega(\lambda\alpha+(1-\lambda)\alpha,\lambda+(1-\lambda)(\beta+\gamma-1),\lambda(\beta+\gamma-1)+(1-\lambda)) \\ &\le \omega(\lambda\alpha,\lambda,\lambda(\beta+\gamma-1))+\omega((1-\lambda)\alpha,(1-\lambda)(\beta+\gamma-1),(1-\lambda)) \\ &= \lambda\cdot\omega(\alpha,1,\beta+\gamma-1)+(1-\lambda)\cdot\omega(\alpha,\beta+\gamma-1,1) \\ &= \omega(\alpha,1,\beta+\gamma-1)\ ,\end{aligned}$$

where the inequality uses Proposition 6, Item 3, the following equality uses Proposition 6, Item 2, and the last one uses Proposition 6, Item 4. ◀

2.3 Input-Sparse Matrix Multiplication

In this paper, we assume *sparse representations* of vectors and matrices. Specifically, we represent a vector $\boldsymbol{x}\in R^n$ by a list of index-value pairs of its non-zero entries. We represent a matrix $A=(\boldsymbol{a}_1,\ldots,\boldsymbol{a}_n)\in R^{m\times n}$ using lists of the non-zero entries of each column $\boldsymbol{a}_i$, sorted by i.

We next give a basic algorithm for matrix multiplication when one of the matrices is column-sparse. This algorithm is folklore, and appears in work at least as early as Gustavson [18].

▶ **Proposition 9** (Naïve Input-Sparse Matrix Multiplication). *Let* $A\in R^{m\times n}, B\in R^{n\times p}$ *for a ring* R *and* $m,n,p\in\mathbb{Z}^+$. *Let* A *be* t*-column-sparse for* $t\in\mathbb{Z}^+$. *Then* $\|AB\|_0\le t\cdot\|B\|_0$, *and there is a deterministic algorithm that computes* AB *using* $t\cdot\|B\|_0\cdot\mathrm{poly}(\log n)$ *arithmetic operations over* R.

Proof. For $i\in[n]$, let b_i be the number of non-zero elements in the ith row of B. If we consider the multiplication of A and B as a sum of outer products, then for every $k\in[n]$, the outer product of the kth column of A and the kth row of B can be computed using at most $t\cdot b_k$ multiplications, and has sparsity at most $t\cdot b_k$. Therefore, $\|AB\|_0\le\sum_{i\in[n]}t\cdot b_k=t\cdot\|B\|_0$. The running time of this algorithm is bounded by $t\cdot\|B\|_0\cdot\mathrm{poly}(\log n)$, where the $\mathrm{poly}(\log n)$ factor accounts for the sparse representation. ◀

Next, we present known bounds on the complexity of matrix multiplication when the input matrices are sparse. The following bound was proven for square matrices in [36], and was generalized to rectangular matrices in [22] (see also [1, Lemma 4.2]).

▶ **Theorem 10** ([36, 22]). *Let $\alpha, \beta, \gamma > 0$ and $0 \le \delta_{in} \le \max\{\alpha+\beta, \beta+\gamma\}$. Then*

$$\omega_{\delta_{in}}(\alpha,\beta,\gamma) \le \min_{\eta\in[0,\beta]} \max\{\omega(\alpha,\eta,\gamma),\, 2\delta_{in}-\eta\}\ .$$

Next, we list two additional bounds on sparse rectangular matrix multiplication exponents $\omega_{\delta_{\text{in}}}(\cdot,\cdot,\cdot)$.

▶ **Proposition 11.** *Let $\alpha, \beta, \gamma > 0$. The following hold:*

1. *If $0 \le \delta_{in} \le \max\{\alpha+\beta, \beta+\gamma\}$, then $\omega_{\delta_{in}}(\alpha,\beta,\gamma) \le \delta_{in} + \min\{\alpha,\gamma\}$.*
2. *If $\delta_{in} \ge \max\{\alpha,\gamma\}$, then $\omega_{\delta_{in}}(\alpha,\beta,\gamma) \ge \alpha+\gamma$.*

Proof. The first bound follows from Proposition 9 by noting that every $n^\alpha \times n^\beta$ matrix is n^α-column-sparse (and that the transpose of an $n^\beta \times n^\gamma$ matrix is n^γ-column-sparse).

For the second bound, consider $A = (\mathbf{1}, \mathbf{0}, \ldots, \mathbf{0}) \in R^{n^\alpha \times n^\beta}$ and $B = (\mathbf{1}, \mathbf{0}, \ldots, \mathbf{0})^T \in R^{n^\beta \times n^\gamma}$. Note that $\|A\|_0, \|B\|_0 \le O(n^{\delta_{\text{in}}})$, and $\|AB\|_0 = n^{\alpha+\gamma}$. Therefore, any algorithm computing AB must have running time at least $\Omega(n^{\alpha+\gamma})$. ◀

3 Output-Sparse Matrix Multiplication

We now present our main technical results on OSMM. In Section 3.1, we formally define compressed sensing schemes and state our main theorem about a concrete such scheme. In Section 3.2, we give a reduction from δ-OSMM to compressed sensing and instantiate the reduction to give our roughly $n^{\omega(\delta/2,1,1)}$-time deterministic algorithm for δ-OSMM, proving Theorem 1. In Section 3.3, we introduce a variant of matrix multiplication verification (MMV) and give a reduction from OSMM to compressed sensing and this MMV variant. We then instantiate the reduction to give our roughly $n^{\omega(\delta-1,1,1)}$-time randomized algorithm for δ-OSMM, proving Theorem 2.

3.1 Compressed Sensing

We will use the following formalization of a compressed sensing scheme.

▶ **Definition 12.** *Let R be a ring and let $m = m(n,t)$ be a non-decreasing, integer-valued function. An m-*compressed sensing scheme *over R is a pair of algorithms $(\mathcal{M}, \mathcal{R})$ that work as follows. On inputs n and t in unary with $1 \le t \le n$, $\mathcal{M}$ outputs a matrix $H \in R^{m\times n}$. On input $H\boldsymbol{x}$ for a vector $\boldsymbol{x} \in R^n$ satisfying $\|\boldsymbol{x}\|_0 \le t$, $\mathcal{R}$ outputs $\boldsymbol{x}$.*[9]

Here the matrix H output by $\mathcal{M}$ is called a *measurement matrix*, and $\mathcal{R}$ is called a *sparse recovery algorithm*. We will crucially use the following theorem about an explicit compressed sensing scheme from [7] (which we instantiate with a construction of expanders from [17]) in both of our main algorithms. The claim in Theorem 13 follows implicitly by combining [7, 17], but we include a full proof for completeness. However, the proof is somewhat involved and so we defer it to an appendix of the full version of this paper [6].

[9] In fact, the sparse recovery algorithm $\mathcal{R}$ needs to know the value of n, and often it also needs t and H as well. This is usually handled by having the algorithm $\mathcal{M}$ also compute a string σ as preprocessing that is passed as auxiliary input to $\mathcal{R}$. Without loss of generality, we assume that σ encodes H, n, and t, but it could also encode more. So, formally we should write $\mathcal{R}(H\boldsymbol{x}, \sigma)$ for running $\mathcal{R}$ on $H\boldsymbol{x}$, but for conciseness we often abuse notation and simply write $\mathcal{R}(H\boldsymbol{x})$.

▶ **Theorem 13** ([7]). *Let R be a ring and let $\alpha > 0$ be a constant. There is a deterministic m-compressed sensing scheme $(\mathcal{M}, \mathcal{R})$ over R with $m = m(n,t) = t^{1+\alpha} \cdot \mathrm{poly}(\log n)$. The algorithm $\mathcal{M}$ runs in time $\widetilde{O}(n)$, and the algorithm $\mathcal{R}$ performs $t^{1+\alpha} \cdot \mathrm{poly}(\log n)$ arithmetic operations over R.*

Furthermore, the measurement matrix $H := \mathcal{M}(1^n, 1^t)$ is binary and d-column-sparse for $d = \mathrm{poly}(\log n)$, and therefore can be multiplied by an arbitrary vector $\boldsymbol{x} \in R^n$ using $\|\boldsymbol{x}\|_0 \cdot \mathrm{poly}(\log n)$ addition operations over R.[10]

3.2 Deterministic OSMM via a Reduction to Compressed Sensing

In this section, we start by giving our two-pass reduction from OSMM to compressed sensing in Algorithm 1. One may also view this reduction as a meta-algorithm. Making it into an explicit algorithm requires using an explicit compressed sensing scheme (we note this in the "Subroutine" heading in the specification of Algorithm 1).

Algorithm 1 Deterministic output-sparse matrix multiplication over a ring R.

Input: Matrices $A, B \in R^{n \times n}$ over a ring R, and $t \in \mathbb{Z}^+$ such that $\|AB\|_0 \le t^2$.
Subroutine: An m-compressed sensing scheme $(\mathcal{M}, \mathcal{R})$ over R.
Output: The product $C = AB$.

1 Compute a measurement matrix $H \in R^{m \times n} := \mathcal{M}(1^n, 1^t)$.
2 Compute $D = (\boldsymbol{d}_1, \ldots, \boldsymbol{d}_n) := HAB$ using fast rectangular MM.
3 Compute $D' := (\mathcal{R}(\boldsymbol{d}_1), \ldots, \mathcal{R}(\boldsymbol{d}_n))$. // t-sparse columns of $C = AB$ agree with D'.
4 Compute $F = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_n) := (H(AB - D')^T)$ using fast rectangular MM.
5 Compute $F' := (\mathcal{R}(\boldsymbol{f}_1), \ldots, \mathcal{R}(\boldsymbol{f}_n))$. // $F' = (AB - D')^T$.
6 **return** $D' + (F')^T$.

We next analyze Algorithm 1.

▶ **Theorem 14.** *Let R be a ring, and let $A, B \in R^{n \times n}$ be matrices satisfying $\|AB\|_0 \le t^2$ for $t \in \mathbb{Z}^+$ with $t^2 \le O(n^\delta)$ for some $\delta \in [0,2]$. Let $(\mathcal{M}, \mathcal{R})$ be an m-compressed sensing scheme over R with $m = m(n,t) \le O(t \cdot n^\alpha)$ for some $\alpha > 0$. Then on input A, B, Algorithm 1 outputs the matrix product $C := AB$. Furthermore, if $\mathcal{M}$ runs in time $T_{\mathcal{M}}(n,t)$ and $\mathcal{R}$ runs in time $T_{\mathcal{R}}(n,t)$ then for any constant $\varepsilon > 0$, Algorithm 1 performs*

$$O(T_{\mathcal{M}}(n,t) + n \cdot T_{\mathcal{R}}(n,t) + n^{\omega(\delta/2+\alpha,1,1)+\varepsilon})$$

arithmetic operations over R.

Proof. We start by proving correctness. Let $C = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n)$, $D' = (\boldsymbol{d}'_1, \ldots, \boldsymbol{d}'_n)$, and $F' = (\boldsymbol{f}'_1, \ldots, \boldsymbol{f}'_n)$. We assume without loss of generality that $\mathcal{R}$ outputs *some* vector in R^n on every input $\boldsymbol{y} \in R^m$ (including when $\boldsymbol{y} \neq H\boldsymbol{x}$ for any t-sparse $\boldsymbol{x} \in R^n$).

We note that C has at most t columns $\boldsymbol{c}_i$ that are not t-sparse, since otherwise we would have $\|C\|_0 > t^2$. Furthermore, by the specification of $\mathcal{R}$, $\boldsymbol{d}'_i = \mathcal{R}(H\boldsymbol{c}_i) = \boldsymbol{c}_i$ for each $i \in [n]$ such that $\boldsymbol{c}_i$ is t-sparse. It follows that every row of $C - D'$ is t-sparse, and therefore that every column of $(AB - D')^T$ is t-sparse. From this and again using the specification of $\mathcal{R}$, we get that $F' = (AB - D')^T$. Correctness then follows by noting that $C = AB = D' + (F')^T$.

[10] Assuming that H and $\boldsymbol{x}$ are given in sparse representation by lists of non-zero entries (as in the output of the algorithm $\mathcal{M}$).

We next analyze the time complexity of Algorithm 1. It makes one call to $\mathcal{M}$ on input $(1^n, 1^t)$, which takes time at most $T_{\mathcal{M}}(n,t)$, and $2n$ calls to $\mathcal{R}$, which takes time at most $2n \cdot T_{\mathcal{R}}(n,t)$. Furthermore, because $H \in R^{m\times n}$ for $t = O(n^{\delta/2})$ and $m \le O(t \cdot n^{\alpha}) = O(n^{\delta/2+\alpha})$, HAB and $H(AB - D')^T$ can be computed in time $O(n^{\omega(\delta/2+\alpha,1,1)+\varepsilon})$ for any constant $\varepsilon > 0$ using fast rectangular matrix multiplication. All other operations run in $O(n^2) \le O(n^{\omega(\delta/2+\alpha,1,1)})$ time. The result follows by summing the running time bounds for each of these components. ◀

Our main theorem about solving OSMM deterministically follows as a corollary.

▶ **Theorem 1** (Deterministic OSMM). *Let $\delta \in [0,2]$, let R be a ring, and let $\varepsilon > 0$ be a constant. There is a deterministic algorithm for solving δ-OSMM$_R$ on $n \times n$ matrices that performs $O(n^{\omega(\delta/2,1,1)+\varepsilon})$ operations over R.*

Proof. We instantiate the compressed sensing scheme subroutine needed in Algorithm 1 with the scheme described in Theorem 13. Let $\alpha > 0$ be a constant, which we will set later, and let α' be a constant satisfying $0 < \alpha' < \alpha$. From Theorem 13, we have that there exists a deterministic m-compressed sensing scheme where $m = m(n,t) = t^{1+\alpha'} \cdot \mathrm{poly}(\log n) \le O(tn^{\alpha})$, with $T_{\mathcal{M}}(n,t) \le T_{\mathcal{M}}(n,n) \le \widetilde{O}(n) \le O(n^{1+\alpha})$ and $T_{\mathcal{R}}(n,t) \le T_{\mathcal{R}}(n,n) \le O(n^{1+\alpha})$. Furthermore, we have that $\omega(\delta/2+\alpha,1,1) \le \omega(\delta/2,1,1) + \omega(\alpha,0,0) = \omega(\delta/2,1,1) + \alpha$ by Proposition 6, Items 1 and 3. Using the fact that $\omega(\delta/2,1,1) \ge 2$ for all $\delta \in [0,2]$, we then have that for any constant $\varepsilon' > 0$ the algorithm performs at most

$$\begin{aligned} O(T_{\mathcal{M}}(n,t) + n \cdot T_{\mathcal{R}}(n,t) + n^{\omega(\delta/2+\alpha,1,1)+\varepsilon'}) &\le O(n^{1+\alpha} + n^{2+\alpha} + n^{\omega(\delta/2,1,1)+\alpha+\varepsilon'}) \\ &\le O(n^{\omega(\delta/2,1,1)+\alpha+\varepsilon'}) \end{aligned}$$

operations over R. To prove the claim, we need to show that the algorithm performs at most $O(n^{\omega(\delta/2,1,1)+\varepsilon})$ operations. This follows by choosing $\varepsilon', \alpha > 0$ such that $\varepsilon' + \alpha \le \varepsilon$. ◀

3.3 Randomized OSMM via a Reduction to a Variant of MMV and Compressed Sensing

In this section, we get a roughly $n^{\omega(\delta-1,1,1)}$-time algorithm for δ-OSMM by reducing to a variant of MMV and to compressed sensing.

3.3.1 Column-Wise Matrix Multiplication Verification

In this section, we introduce and give an algorithm for the variant of MMV in which the goal is to check whether each column of AB is equal to its corresponding column in C'.

▶ **Definition 15.** *For a ring R, the Column-Wise MMV problem over R (Column-Wise-MMV$_R$) is defined as follows. The input consists of matrices $A \in R^{m\times n}$, $B \in R^{n\times p}$, and $C' = (\boldsymbol{c}'_1, \ldots, \boldsymbol{c}'_p) \in R^{m\times p}$. Let $C = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_p) := AB$. The goal is to output the set $J \subseteq [p]$ of indices of columns on which C and C' differ, i.e.,*

$$J := \{j \in [p] : \boldsymbol{c}_j \neq \boldsymbol{c}'_j\} \ .$$

We note that "normal" MMV corresponds to deciding whether $J = \emptyset$ in Definition 15, and so MMV easily reduces to Column-Wise-MMV. We next argue that a slight variant of Freivalds's algorithm [12] also solves Column-Wise-MMV in nearly quadratic time.

▶ **Lemma 16** (Freivalds's algorithm for Column-Wise-MMV)**.** *Let R be a ring and let $c \in \mathbb{Z}^+$ be a constant. For any $m, n, p \in \mathbb{Z}^+$, there is a randomized algorithm that solves* Column-Wise-MMV$_R$ *on matrices $A \in R^{m\times n}, B \in R^{n\times p}, C' \in R^{m\times p}$ performing $O(\max\{mn, np\} \cdot \log(mnp))$ arithmetic operations over R and has a success probability of at least $1 - 1/(mnp)^c$. In particular, when $m, p = O(n)$, the algorithm performs $O(n^2 \log n)$ arithmetic operations over R and has a success probability of at least $1 - 1/n^c$.*

Proof. The algorithm works as follows. First, it samples a uniformly random matrix $X \sim \{0,1\}^{N\times m}$ for $N := \lceil (c+1)\log(mnp)\rceil$. It then computes $D = (\boldsymbol{d}_1, \ldots, \boldsymbol{d}_p) := X(AB - C')$, and outputs $J' := \{i \in [p] : \boldsymbol{d}_i \neq \boldsymbol{0}\}$.

We start by showing correctness of the algorithm, i.e., that $J' = J$ for J as defined in Definition 15. For a fixed non-zero vector $\boldsymbol{y} \in R^m$ and a uniformly random $\boldsymbol{x} \sim \{0,1\}^m$, $\Pr[\langle \boldsymbol{x}, \boldsymbol{y}\rangle \neq 0] \geq 1/2$. Indeed, this follows by noting that each term $x_i y_i$ in $\langle \boldsymbol{x}, \boldsymbol{y}\rangle = \sum_{i=1}^m x_i y_i$ for non-zero y_i is equal to 0 with probability $1/2$ and an induction argument. So, for any non-zero column $\boldsymbol{y}_i$ of $AB - C'$, we get that $\Pr[X\boldsymbol{y}_i = \boldsymbol{0}] \leq 1/2^N$. (If $\boldsymbol{y}_i = 0$, then $X\boldsymbol{y}_i = \boldsymbol{0}$ with probability 1.) Taking a union bound over the at most p non-zero columns of $AB - C'$, we then get that the probability that there exists a non-zero column $\boldsymbol{y}_i$ of $AB - C'$ such that $\boldsymbol{d}_i = \boldsymbol{0}$ is at most $p/2^N \leq p/(mnp)^{c+1} \leq 1/(mnp)^c$, as needed.

Furthermore, the algorithm runs in $O(N \cdot \max\{mn, np\}) = O(\max\{mn, np\} \cdot \log(mnp))$ time, which finishes the analysis. ◀

3.3.2 The Reduction and its Analysis

We next give our reduction from OSMM (and in fact, simply MM) to Column-Wise-MMV and compressed sensing in Algorithm 2. We sometimes use sets to index the rows and columns of a matrix. I.e., for finite sets I and J we use notation like $A \in R^{I\times J}$ to denote a matrix over a ring R with $|I|$ rows indexed by I and $|J|$ columns indexed by J. Similarly, we use notation like $A_{I',J'}$ to denote the submatrix of $A \in R^{I\times J}$ with rows indexed by $I' \subseteq I$ and columns indexed by $J' \subseteq J$.

In Theorem 17, we analyze Algorithm 2. We again emphasize that its correctness does not depend on the sparsity of AB; only its running time does. Furthermore, the same algorithm works regardless of the sparsity of AB.

▶ **Theorem 17.** *Let $A, B \in R^{n\times n}$ over a ring R, let $(\mathcal{M}, \mathcal{R})$ be an m-compressed sensing scheme with $m = m(n,t) \leq O(tn^\alpha)$ for some $\alpha \in (0, \omega^\perp)$, and let* ColMMV *be an algorithm for* Column-Wise-MMV *over R. Then on input $A, B \in R^{n\times n}$, Algorithm 2 outputs the matrix product $C = AB$.*

Furthermore, let $T_{\mathcal{M}}(n,t)$ and $T_{\mathcal{R}}(n,t)$ denote the running times of $\mathcal{M}$ and $\mathcal{R}$, respectively, let $T_H(n,t)$ denote the time needed to multiply $H := \mathcal{M}(1^n, 1^t)$ by an $n \times n$ matrix, and let $T_V(m,n,p)$ denote the running time of ColMMV *on input matrices $A \in R^{m\times n}, B \in R^{n\times p}, C \in R^{m\times p}$. If $\|AB\|_0 \leq O(n^\delta)$ for $0 \leq \delta \leq 2$ then the algorithm performs*

$$\widetilde{O}(n^{\beta+\varepsilon} + T_{\mathcal{M}}(n,n) + n \cdot T_{\mathcal{R}}(n,n) + T_H(n,n) + T_V(n,n,n))$$

arithmetic operations over R for any constant $\varepsilon > 0$, where

$$\beta := \begin{cases} 2 & \text{if } \delta + \alpha \leq 1 + \omega^\perp \text{ ,} \\ \omega(\delta + \alpha - 1, 1, 1) & \text{otherwise .} \end{cases}$$

Proof. Assume without loss of generality that $\delta \geq 1$. If not, set $\delta = 1$, and using that $\alpha \leq \omega^\perp$ conclude that $\beta = 2$. Let $C = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n)$.

Algorithm 2 Matrix multiplication using column-wise matrix multiplication verification.

Input: Matrices $A, B \in R^{n\times n}$ over a ring R.
Subroutine: An m-compressed sensing scheme $(\mathcal{M}, \mathcal{R})$ over R.
Subroutine: An algorithm ColMMV for Column-Wise-MMV.
Output: The product AB.

Set $J_0 := [n]$.
Initialize a matrix $C' = (\boldsymbol{c}'_1, \ldots, \boldsymbol{c}'_n) \in R^{n\times n}$ arbitrarily.
for $i = 0, 1, \ldots, \lceil \log n \rceil$ **do**
  Set $t := 2^i$.
  Compute $H := \mathcal{M}(1^n, 1^t) \in R^{m\times n}$. // $m = m(n,t)$.
  Compute $B' := B_{[n],J_i} \in R^{[n]\times J_i}$. // Restrict B to columns indexed by J_i.
  Compute $D = (\boldsymbol{d}_j)_{j\in J_i} := HAB' \in R^{[m]\times J_i}$.
  Compute $F = (\boldsymbol{f}_j)_{j\in J_i} := (\mathcal{R}(\boldsymbol{d}_j))_{j\in J_i} \in R^{[n]\times J_i}$.
  Compute $J_{i+1} := \text{ColMMV}(A, B', F)$. // Indices of columns that differ in AB', F.
  for $j \in J_i \setminus J_{i+1}$ **do**
    Set $\boldsymbol{c}'_j := \boldsymbol{f}_j$.
return C'

We argue that after the ith iteration of the for loop, $\boldsymbol{c}'_j = \boldsymbol{c}_j$ for all $j \in [n]$ such that $\|\boldsymbol{c}_j\|_0 \le 2^i$. Indeed, this follows from the fact that for such an index j, $\boldsymbol{f}_j = \mathcal{R}(H\boldsymbol{c}_j) = \boldsymbol{c}_j$ by the definition of a compressed sensing scheme with $t = 2^i$, and therefore we have that $j \notin J_{i+1}$. (As in the proof of Theorem 14 we assume that $\mathcal{R}$ always outputs *some* vector in R^n, regardless of its input.) In particular, because $\|\boldsymbol{c}_j\|_0 \le n \le 2^{\lceil \log n \rceil}$ for all columns $\boldsymbol{c}_j$, $\boldsymbol{c}'_j = \boldsymbol{c}_j$ for all $j \in [n]$ after the last iteration of the for loop, which implies that $C' = C$, as needed.

We next turn to analyzing the algorithm's running time. We upper bound the worst-case running time of a single iteration (iteration i) of the for loop, and note that the loop performs $O(\log n)$ iterations. The $\mathcal{M}(1^n, 1^t)$ call takes time $T_{\mathcal{M}}(n,t) \le T_{\mathcal{M}}(n,n)$, the $|J_i| \le n$ calls to $\mathcal{R}$ take time at most $|J_i| \cdot T_{\mathcal{R}}(n,t) \le n \cdot T_{\mathcal{R}}(n,n)$, and the call to ColMMV takes time at most $T_V(n, n, |J_i|) \le T_V(n,n,n)$.

It remains to upper bound the complexity of computing HAB' in the ith iteration, which we compute parenthesized as $(HA)B'$. Computing $A' := HA$ takes $T_H(n,t) \le T_H(n,n)$ time by assumption,[11] and it remains to compute $A'B'$, which we do using rectangular matrix multiplication. For $i = 0$, we have $t = 1$ and $m = m(n,t) = O(n^\alpha)$. Therefore, computing $A'B'$ takes $O(n^{\omega(\alpha,1,1)+\varepsilon}) = O(n^{\omega(\delta+\alpha-1,1,1)+\varepsilon})$ time. For $i \ge 1$, we have $t = 2^i$ and $(t/2) \cdot |J_i| \le \|AB\|_0$, which holds because J_i consists of indices of columns of AB with more than $t/2$ non-zero entries.

Let $\eta, \gamma \ge 0$ be such that $t = O(n^\eta)$ and $|J_i| = O(n^\gamma)$. Then because $A' \in R^{m\times n}$ for $m = m(n,t) = O(tn^\alpha) = O(n^{\eta+\alpha})$ and $B' \in R^{n\times|J_i|}$, $A'B'$ can be computed in $O(n^{\omega(\eta+\alpha,1,\gamma)+\varepsilon})$ time for any constant $\varepsilon > 0$. Furthermore, since $(t/2) \cdot |J_i| \le \|AB\|_0 \le O(n^\delta)$, we have that $\eta + \gamma \le \delta$. If $\eta + \alpha + \gamma \le 1$, then by Proposition 6, Item 1, $\omega(\eta+\alpha, 1, \gamma) \le \eta + \alpha + 1 + \gamma \le 2$. Otherwise, if $\eta + \alpha + \gamma > 1$, then by Proposition 6, Item 4 and Corollary 8, $\omega(\eta+\alpha, 1, \gamma) \le \omega(\eta+\alpha+\gamma-1, 1, 1) \le \omega(\delta+\alpha-1, 1, 1)$, as needed. ◀

[11] When instantiating this reduction each matrix H will be sparse. Such matrices admit fast multiplication.

Our main theorem about solving OSMM via a randomized algorithm follows as a corollary.

▶ **Theorem 2** (Randomized OSMM). *Let $\delta \in [0,2]$, let R be a ring, and let $\varepsilon > 0$ be a constant. There is a randomized algorithm for solving δ-OSMM$_R$ on $n \times n$ matrices that performs $O(n^{\beta+\varepsilon})$ operations over R, where*

$$\beta := \begin{cases} 2 & \text{if } \delta \leq 1 + \omega^{\perp} \ , \\ \omega(\delta - 1, 1, 1) & \text{otherwise} \ . \end{cases}$$

Proof. Assume without loss of generality that $\delta \geq 1$. If not, set $\delta = 1$. We instantiate the compressed sensing scheme subroutine needed in Algorithm 2 with the scheme described in Theorem 13. The running time analysis follows from a similar argument as in the proof of Theorem 1, combined with the guarantee that $T_H(n,n) \leq \widetilde{O}(n^2)$ from Theorem 13 and the guarantee that $T_V(n,n,n) \leq \widetilde{O}(n^2)$ from Lemma 16. In particular, the fact that H can be multiplied by an arbitrary vector $\boldsymbol{x}$ in $\widetilde{O}(\|\boldsymbol{x}\|_0)$ time implies that H can be multiplied by an arbitrary matrix $A \in R^{n \times n}$ in $\widetilde{O}(n^2)$ time. This implies that $T_H(n,n) \leq \widetilde{O}(n^2)$.

Furthermore, the algorithm in Lemma 16 is randomized and succeeds with probability $1 - 1/n^c$ for any constant $c > 0$. Fix any such $c > 0$. Algorithm 2 makes $O(\log n)$ calls to ColMMV, and so by taking a union bound we get that all of these calls succeed with probability at least $1 - O(\log n / n^c) = 1 - o(1)$. It follows that Algorithm 2 succeeds with probability $1 - o(1)$, as needed. ◀

4 Fully Sparse Matrix Multiplication

In this section, we describe how to extend our main algorithms for OSMM to the fully sparse setting when the input matrices A, B are promised not only to satisfy $\|AB\|_0 \leq O(n^{\delta_{\text{out}}})$ but also $\|A\|_0, \|B\|_0 \leq O(n^{\delta_{\text{in}}})$ for some $\delta_{\text{in}} \in [0,2]$.

4.1 Deterministic FSMM

We first give our deterministic algorithm for FSMM, which follows by instantiating Algorithm 1 with the compressed sensing scheme in Theorem 13 as in the proof of Theorem 1.

▶ **Theorem 18** (Deterministic FSMM). *Let $\delta_{in}, \delta_{out} \in [0,2]$, let R be a ring, and let $\varepsilon > 0$ be a constant. There is a deterministic algorithm for solving $(\delta_{in}, \delta_{out})$-FSMM$_R$ on $n \times n$ matrices that performs $O(n^{\beta+\varepsilon})$ operations over R, where*

$$\beta := \max\{\delta_{out}/2 + 1, \omega_{\delta_{in}}(\delta_{out}/2, 1, 1)\} \ .$$

Proof sketch. We refer to Algorithm 1, instantiated with the compressed sensing scheme in Theorem 13, and show how to implement its operations under the assumption that $\|A\|_0, \|B\|_0 \leq O(n^{\delta_{\text{in}}})$. Let $\alpha > 0$ be a small constant to be set in the analysis. We first note that $T_{\mathcal{M}}(n,t) \leq \widetilde{O}(n)$ as before.

We now analyze the time needed to compute $D = HAB$. By Theorem 13, H is poly$(\log n)$-column-sparse, and so it is possible to compute $A' := HA$ in $\widetilde{O}(n^{\delta_{\text{in}}})$ time by Proposition 9. Since $\|A'\|_0 = \widetilde{O}(n^{\delta_{\text{in}}})$ and $H, A' \in R^{m \times n}$ for $m = O(n^{\delta_{\text{out}}/2+\alpha})$, it is possible to compute the product $A'B$ in $O(n^{\omega_{\delta_{\text{in}}}(\delta_{\text{out}}/2,1,1)+\varepsilon})$ time for any constant $\varepsilon > 0$ (assuming that $\alpha > 0$ is chosen to be sufficiently small; see the proof of Theorem 1). We note that $\omega_{\delta_{\text{in}}}(\delta_{\text{out}}/2, 1, 1) \geq \delta_{\text{in}}$ (since simply reading in an $n \times n$ matrix B with $\Theta(n^{\delta_{\text{in}}})$ non-zero entries takes $\Omega(n^{\delta_{\text{in}}})$ time), and so computing HAB takes $O(n^{\omega_{\delta_{\text{in}}}(\delta_{\text{out}}/2,1,1)+\varepsilon})$ time overall.

Furthermore, by Theorem 13, $T_{\mathcal{R}}(t,n) \leq t^{1+\alpha} \cdot \text{poly}(\log n) \leq \widetilde{O}(n^{\delta_{\text{out}}/2+\alpha})$ since $t = O(n^{\delta_{\text{out}}/2})$, and so computing D' and F' takes $\widetilde{O}(n^{\delta_{\text{out}}/2+\alpha+1})$ time. Finally, we analyze the time needed to compute F. Computing $H(AB)^T$ can be done in $O(n^{\omega_{\delta_{\text{in}}}(\delta_{\text{out}}/2,1,1)+\varepsilon})$ time for any constant $\varepsilon > 0$ as with computing D. Furthermore, D' has at most $O(n^{\delta_{\text{out}}/2})$ non-zero columns, and so, again using the fact that H is $\text{poly}(\log n)$-column-sparse, it takes $\widetilde{O}(n^{\delta_{\text{out}}/2+1})$ time to compute $H(D')^T$. The claim follows. ◀

We recover Theorem 1 from Theorem 18 by noting that $\omega_2(\delta_{\text{out}}/2,1,1) = \omega(\delta_{\text{out}}/2,1,1)$ and that $\omega(\delta_{\text{out}}/2,1,1) \geq \delta_{\text{out}}/2+1$ (by Proposition 6, Item 1).

Using the bounds on $\omega_{\delta_{\text{in}}}(\delta_{\text{out}}/2,1,1)$ from Proposition 11, Item 1 and Theorem 10, we conclude that for every $\varepsilon > 0$, $(\delta_{\text{in}}, \delta_{\text{out}})$-FSMM$_R$ can be solved using $O(n^{\delta_{\text{out}}/2+1+\varepsilon} + n^{\delta_{\text{out}}/2+\delta_{\text{in}}+\varepsilon})$ or $O(n^{\delta_{\text{out}}/2+1+\varepsilon} + n^{\sigma+\varepsilon})$ operations over R, where

$$\sigma \leq \min_{\eta \in [0,1]} \max\{\omega(\delta_{\text{out}}/2, \eta, 1),\ 2\delta_{\text{in}} - \eta\}\ . \tag{2}$$

4.2 Randomized FSMM

We now give our randomized algorithm for FSMM as Theorem 19. It follows by instantiating Algorithm 2 with the compressed scheme in Theorem 13 as in the proof of Theorem 2. We recall that Theorem 19 is equivalent to [1, Lemma 3.1], although the algorithm that it is based on uses different techniques.

▶ **Theorem 19** (Randomized FSMM). *Let $\delta_{in} \in [1,2], \delta_{out} \in [0, 2\delta_{in}]$, let R be a ring, and let $\varepsilon > 0$ be a constant. There is a randomized algorithm for solving $(\delta_{in}, \delta_{out})$-FSMM$_R$ on $n \times n$ matrices that performs $O(n^{\delta_{in}+\varepsilon} + n^{\beta+\varepsilon})$ operations over R, where*

$$\beta := \max_{\substack{\eta_1,\eta_2 \in [0,1]\,, \\ \eta_1+\eta_2=\delta_{out}}} \omega_{\delta_{in}}(\eta_1, 1, \eta_2)\ .$$

Proof sketch. We refer to Algorithm 2, instantiated with the compressed sensing scheme in Theorem 13, and show how to implement its operations under the assumption that $\|A\|_0, \|B\|_0 \leq O(n^{\delta_{\text{in}}})$.

As in the proof of Theorem 2, we upper bound the worst-case running time of a single iteration (iteration i) of the for loop for $i \geq 1$ (as the running time for $i = 0$ does not affect the final bound). Furthermore, as noted there, we have that $t \cdot |J_i| = O(n^{\delta_{\text{out}}})$ at each iteration of the loop. I.e., choosing $\eta_1 > 0$ so that $t = 2^i = O(n^{\eta_1})$, we have that $|J_i| \leq O(n^{\eta_2})$ where $\eta_2 \leq \min\{1, \delta_{\text{out}} - \eta_1\}$. We then have that $m = O(tn^{\alpha}) = O(n^{\eta_1+\alpha})$ for an arbitrarily small constant $\alpha > 0$ to be set in the analysis. We note that $T_{\mathcal{M}}(n,t) \leq \widetilde{O}(n)$. Moreover, we have that computing F takes $|J_i| \cdot T_{\mathcal{R}}(n,t) \leq n^{\delta_{\text{out}}-\eta_1} \cdot t^{1+\alpha} \cdot \text{poly}(\log n) = \widetilde{O}(n^{\delta_{\text{out}}+\alpha})$ time.

We next sketch how to modify the algorithm in the proof of Lemma 16 to run in $\widetilde{O}(n^{\delta_{\text{in}}})$ time. We note that the matrix X sampled there has $O(\log n)$ rows, and that we need to compute XAB' and XF. Computing XA and $(XA)B'$ takes $\widetilde{O}(n^{\delta_{\text{in}}})$ time by Proposition 9. On the other hand, we have that $\mathcal{R}$ runs in $t^{1+\alpha} \cdot \text{poly}(\log n) = \widetilde{O}(n^{\eta_1+\alpha})$ time, and therefore each of the $|J_i| = O(n^{\delta_{\text{out}}-\eta_1})$ columns of F must be $\widetilde{O}(n^{\eta_1+\alpha})$ sparse. Therefore, we can compute XF in time $\widetilde{O}(n^{\delta_{\text{out}}+\alpha})$ time.

It remains to analyze the time complexity of computing $D = HAB'$, since all other operations are efficient. We have that $A' := HA$ can be computed in $\widetilde{O}(n^{\delta_{\text{in}}})$ time by Theorem 13. Furthermore, since H is $\text{poly}(\log n)$ column sparse, A' is itself $\widetilde{O}(n^{\delta_{\text{in}}})$ sparse. It follows that $HAB' = A'B'$ can also be computed in $O(n^{\omega_{\delta_{\text{in}}}(\eta_1,1,\eta_2)+\varepsilon})$ time for any constant $\varepsilon > 0$.

To finish the proof, we note that the for loop uses at most $O(\log n)$ iterations, and then bound the running time of each iteration. The running time of the ith iteration is bounded from above by $\widetilde{O}(n^{\delta_{\text{out}}+\alpha} + n^{\delta_{\text{in}}} + n^{\omega_{\delta_{\text{in}}}(\eta_1,1,\eta_2)+\varepsilon})$ for arbitrarily small constants α and ε. In particular, to prove the claim, it suffices to show that both δ_{out} and $\omega_{\delta_{\text{in}}}(\eta_1, 1, \eta_2)$ are bounded from above by β. First, by the definition of β, we have that $\omega_{\delta_{\text{in}}}(\eta_1, 1, \eta_2) \leq \beta$.

Second, by Proposition 11, Item 2, we have that

$$\beta \geq \omega_{\delta_{\text{in}}}(\eta^*, 1, \delta_{\text{out}} - \eta^*) \geq \delta_{\text{out}} \ ,$$

where $\eta^* := \min\{1, \delta_{\text{out}}\}$. Indeed, Proposition 11, Item 2 applies because of the following bounds. First, $\eta^* = \min\{1, \delta_{\text{out}}\} \leq 1 \leq \delta_{\text{in}}$. Second, if $\delta_{\text{out}} < 1$, then $\delta_{\text{out}} - \eta^* = \delta_{\text{out}} - \delta_{\text{out}} = 0 \leq \delta_{\text{in}}$. Third, if $\delta_{\text{out}} \geq 1$, then $\delta_{\text{out}} - \eta^* = \delta_{\text{out}} - 1 \leq 2 - 1 = 1 \leq \delta_{\text{in}}$. Here we have repeatedly used the assumption that $\delta_{\text{in}} \geq 1$. ◀

We recover Theorem 2 from Theorem 19 by setting $\delta_{\text{in}} = 2$ and noting that for $\delta_{\text{out}} \geq 1$,

$$\max_{\substack{\eta_1,\eta_2\in[0,1]\,,\\ \eta_1+\eta_2=\delta_{\text{out}}}} \omega_{\delta_{\text{in}}}(\eta_1, 1, \eta_2) = \max_{\substack{\eta_1,\eta_2\in[0,1]\,,\\ \eta_1+\eta_2=\delta_{\text{out}}}} \omega(\eta_1, 1, \eta_2) \leq \omega(\delta_{\text{out}} - 1, 1, 1)$$

by Corollary 8.

While we state Theorem 19 for $\delta_{\text{in}} \geq 1$ for simplicity, we remark that the proof can be extended to the case of $\delta_{\text{in}} \in [0, 2]$. The main challenge in this case is that we cannot afford to run the algorithm $\mathcal{M}$ it time $\widetilde{O}(n)$ anymore (as this bound might be larger than the claimed upper bound on the running time of the algorithm). To overcome this, we note that the construction of the matrix H in our compressed sensing scheme (presented in full in an appendix of the full version of this paper [6]) is in fact strongly explicit. That is, one can compute all non-zero entries of a given column of H, and therefore multiply by H efficiently without even running the algorithm $\mathcal{M}$ first.

4.3 Optimal FSMM (and OSMM)

We now prove that our randomized algorithm for FSMM in Theorem 19 (which is equivalent to [1, Lemma 3.1]) is optimal in the sense that any improvement on it would imply a corresponding improvement on input-sparse rectangular matrix multiplication exponents. In fact, we show an *equivalence* between these exponents and the running time exponent for $(\delta_{\text{in}}, \delta_{\text{out}})$-FSMM$_R$. We also show an analogous result for the optimality of the algorithm in Theorem 2 for OSMM.

▶ **Proposition 20.** *Let $\delta_{in} \in [1, 2]$, $\delta_{out} \in [0, 2\delta_{in}]$, let R be a ring, and let $\beta \geq 0$. Then there is a randomized algorithm for $(\delta_{in}, \delta_{out})$-FSMM$_R$ that runs in time $O(n^{\beta+\varepsilon})$ for every constant $\varepsilon > 0$ if and only if $\beta \geq \max\{\delta_{in}, \omega_{\delta_{in}}(\eta_1, 1, \eta_2)\}$ for all $\eta_1, \eta_2 \in [0, 1]$ with $\eta_1 + \eta_2 = \delta_{out}$.*

Furthermore, if $\delta_{out} \in [1, 2]$ then there is a randomized algorithm for δ_{out}-OSMM that runs in time $O(n^{\beta+\varepsilon})$ for every constant $\varepsilon > 0$ if and only if $\beta \geq \omega(\delta_{out} - 1, 1, 1)$.

Proof. Let $\eta_1, \eta_2 \in [0, 1]$ with $\eta_1 + \eta_2 = \delta_{\text{out}}$. We reduce the problem of computing the product of $A \in R^{m\times n}$ and $B \in R^{n\times p}$ for $m = O(n^{\eta_1})$ and $p = O(n^{\eta_2})$ when A and B are both $O(n^{\delta_{\text{in}}})$-sparse to $(\delta_{\text{in}}, \delta_{\text{out}})$-FSMM$_R$. The reduction simply outputs $A' := (A^T, \mathbf{0}, \ldots, \mathbf{0})^T \in R^{n\times n}$ and $B' := (B, \mathbf{0}, \ldots, \mathbf{0}) \in R^{n\times n}$. I.e., A' and B' are simply A and B padded with enough rows and columns of 0s, respectively, to make them square. It is easy to recover the product AB from the product $A'B'$. Furthermore, $\|A'\|_0 = \|A\|_0 = O(n^{\delta_{\text{in}}})$, $\|B'\|_0 = \|B\|_0 = O(n^{\delta_{\text{in}}})$, and, because $AB \in R^{m\times p}$, $\|A'B'\|_0 \leq mp \leq O(n^{\eta_1+\eta_2}) = O(n^{\delta_{\text{out}}})$, as needed. Because the

reduction worked for arbitrary $\eta_1, \eta_2 \in [0,1]$ with $\eta_1 + \eta_2 = \delta_{\text{out}}$, we get that if there is an algorithm using $O(n^{\beta+\varepsilon})$ operations over R for $(\delta_{\text{in}}, \delta_{\text{out}})$-FSMM$_R$ for some $\beta > 0$ and any $\varepsilon > 0$, then $\omega_{\delta_{\text{in}}}(\eta_1, 1, \eta_2) \le \beta$. Furthermore, we trivially have that such β must satisfy $\beta \ge \delta_{\text{in}}$ since it takes $\Omega(n^{\delta_{\text{in}}})$ time to read an $(\delta_{\text{in}}, \delta_{\text{out}})$-FSMM$_R$ instance.

On the other hand, Theorem 19 implies that if $\beta \ge \delta_{\text{in}}$ and $\beta \ge \omega_{\delta_{\text{in}}}(\eta, 1, \delta_{\text{out}} - \eta)$ for all $\eta \in [0, \delta_{\text{out}}]$ then $(\delta_{\text{in}}, \delta_{\text{out}})$-FSMM$_R$ is solvable using $O(n^{\beta+\varepsilon})$ many operations over R.

The result for OSMM follows by combining the padding argument above, instantiated with $\delta_{\text{in}} = 2$ and $\eta = 1$, and Theorem 2. ◀

References

1 Amir Abboud, Karl Bringmann, Nick Fischer, and Marvin Künnemann. The time complexity of fully sparse matrix multiplication. In *SODA*, 2024. 1, 2, 3, 4, 5, 7, 10, 17, 18

2 Josh Alman, Ran Duan, Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. More asymmetry yields faster matrix multiplication. In *SODA*, 2025. 2, 9

3 Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. In *SODA*, 2021. 9

4 Rasmus Resen Amossen and Rasmus Pagh. Faster join-projects and sparse matrix multiplications. In *ICDT*, 2009. 3

5 Huck Bennett, Karthik Gajulapalli, Alexander Golovnev, and Evelyn Warton. Matrix multiplication verification using coding theory. In *RANDOM*, 2024. doi:10.4230/LIPIcs.APPROX/RANDOM.2024.42. 4, 6

6 Huck Bennett, Karthik Gajulapalli, Alexander Golovnev, and Evelyn Warton. Output-sparse matrix multiplication using compressed sensing, 2025. Full version. Available at arXiv:2508.10250. doi:10.48550/arXiv.2508.10250. 1, 3, 8, 11, 18

7 Radu Berinde, Anna C. Gilbert, Piotr Indyk, Howard Karloff, and Martin J. Strauss. Combining geometry and combinatorics: A unified approach to sparse signal recovery. In *Allerton*, 2008. 3, 7, 8, 11, 12

8 Michele Borassi, Pierluigi Crescenzi, and Michel Habib. Into the square: On the complexity of some quadratic-time solvable problems. In *ICTCS*, 2015. 3

9 Emmanuel J. Candès, Justin K. Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory*, 52(2):489–509, 2006. doi:10.1109/TIT.2005.862083. 8

10 Shaleen Deep, Xiao Hu, and Paraschos Koutris. Fast join project query evaluation using matrix multiplication. In *SIGMOD*, 2020. 3

11 Ran Duan, Hongxun Wu, and Renfei Zhou. Faster matrix multiplication via asymmetric hashing. In *FOCS*, 2023. 9

12 Rūsiņš Freivalds. Fast probabilistic algorithms. In *MFCS*, 1979. 8, 13

13 Jianhua Gao, Weixing Ji, Fangli Chang, Shiyu Han, Bingxin Wei, Zeming Liu, and Yizhuo Wang. A systematic survey of general sparse matrix-matrix multiplication. *ACM Computing Surveys*, 55(12):1–36, 2023. doi:10.1145/3571157. 3

14 Leszek Gasieniec, Christos Levcopoulos, Andrzej Lingas, Rasmus Pagh, and Takeshi Tokuyama. Efficiently correcting matrix products. *Algorithmica*, 79(2):428–443, 2017. doi:10.1007/S00453-016-0202-3. 3, 6

15 Anna C. Gilbert and Piotr Indyk. Sparse recovery using sparse matrices. *Proceedings of the IEEE*, 98(6):937–947, 2010. doi:10.1109/JPROC.2010.2045092. 8

16 John R. Gilbert, Steve Reinhardt, and Viral B. Shah. High-performance graph algorithms from parallel sparse matrices. In *PARA*, 2006. 3

17 Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. *Journal of the ACM*, 56(4):1–34, 2009. doi:10.1145/1538902.1538904. 1, 3, 7, 8, 11

18 Fred G. Gustavson. Two fast algorithms for sparse matrices: Multiplication and permuted transposition. *ACM Transactions on Mathematical Software*, 4(3):250–269, 1978. doi:10.1145/355791.355796. 3, 10

19 Piotr Indyk. Explicit constructions for compressed sensing of sparse signals. In *SODA*, 2008. 5

20 Mark A. Iwen and Craig V. Spencer. A note on compressed sensing and the complexity of matrix multiplication. *Information Processing Letters*, 109(10):468–471, 2009. doi:10.1016/J.IPL.2009.01.010. 3, 5

21 Riko Jacob and Morten Stöckel. Fast output-sensitive matrix multiplication. In *ESA*, 2015. 3

22 Haim Kaplan, Micha Sharir, and Elad Verbin. Colored intersection searching via sparse rectangular matrix multiplication. In *SoCG*, 2006. 7, 10, 11

23 Marvin Künnemann. On nondeterministic derandomization of Freivalds' algorithm: Consequences, avenues and algorithmic progress. In *ESA*, 2018. doi:10.4230/LIPIcs.ESA.2018.56. 3, 4, 5

24 Konstantin Kutzkov. Deterministic algorithms for skewed matrix products. In *STACS*, 2013. doi:10.4230/LIPIcs.STACS.2013.466. 3, 4, 5

25 François Le Gall. Faster algorithms for rectangular matrix multiplication. In *FOCS*, 2012. 10

26 François Le Gall. Faster rectangular matrix multiplication by combination loss analysis. In *SODA*, 2024. 9

27 François Le Gall and Florent Urrutia. Improved rectangular matrix multiplication using powers of the Coppersmith-Winograd tensor. In *SODA*, 2018. 9

28 Andrzej Lingas. A fast output-sensitive algorithm for boolean matrix multiplication. In *ESA*, 2009. 3

29 Grazia Lotti and Francesco Romani. On the asymptotic complexity of rectangular matrix multiplication. *Theoretical Computer Science*, 23(2):171–185, 1983. doi:10.1016/0304-3975(83)90054-3. 9, 10

30 Rasmus Pagh. Compressed matrix multiplication. In *ITCS*, 2012. 3

31 Rasmus Pagh and Morten Stöckel. The input/output complexity of sparse matrix multiplication. In *ESA*, 2014. 8

32 Daniel S. Roche. What can (and can't) we do with sparse polynomials? In *ISSAC*, 2018. 3

33 Oguz Selvitopi, Md Taufique Hussain, Ariful Azad, and Aydın Buluç. Optimizing high performance Markov clustering for pre-exascale architectures. In *IPDPS*, 2020. 3

34 Dirk Van Gucht, Ryan Williams, David P. Woodruff, and Qin Zhang. The communication complexity of distributed set-joins with applications to matrix multiplication. In *PODS*, 2015. 3

35 Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. New bounds for matrix multiplication: from alpha to omega. In *SODA*, 2024. 3, 5, 9

36 Raphael Yuster and Uri Zwick. Fast sparse matrix multiplication. *ACM Transactions On Algorithms*, 1(1):2–13, 2005. doi:10.1145/1077464.1077466. 3, 7, 10, 11

37 Yudong Zhang, Lenan Wu, Geng Wei, and Shuihua Wang. A novel algorithm for all pairs shortest path problem based on matrix multiplication and pulse coupled neural network. *Digital Signal Processing*, 21(4):517–521, 2011. doi:10.1016/J.DSP.2011.02.004. 3

From Decision to Random Certificates: Exponential Separation for Edge Estimation with Independent Set Queries

Debarshi Chanda
Indian Statistical Institute, Kolkata, India

Buddha Dev Das
Indian Statistical Institute, Kolkata, India

Arijit Ghosh
Indian Statistical Institute, Kolkata, India

Gopinath Mishra
Institute of Mathematical Sciences, Chennai, India

Abstract

We study the problem of estimating the number of edges in an undirected, unweighted graph using sublinear query access. We consider a query model that preserves the structure of Independent Set (IS) queries, but augments their output with a random certificate: given a vertex subset, the oracle returns a uniformly random edge from the induced subgraph if one exists, and returns null otherwise.

Using this access, we give a randomized algorithm that outputs a $(1 \pm \varepsilon)$-approximation to the number of edges with constant success probability using $\widetilde{O}(\log^2 m)$ queries. This implies an exponential separation from both standard IS queries and global random edge-sampling models: estimating the number of edges using standard IS queries require $\widetilde{\Theta}\left(\min\left\{\sqrt{m}, \frac{n}{\sqrt{m}}\right\}\right)$ queries, while direct random edge-sample access requires $\widetilde{\Theta}(\sqrt{m})$ samples. Beyond separation in query complexity, our algorithm is output-sensitive: its query complexity is polylogarithmic in the number of edges in the graph. This aligns with the classical objective in group testing, where one seeks algorithms that are both worst-case optimal and instance-adaptive.

Conceptually, our model connects group testing, the decision–versus–counting dichotomy, graph property testing, and the "power of a random certificate", and can be viewed as a structured form of conditional sampling of edges in graphs.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases Property Testing, Edge Estimation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.41

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2607.07483` [20]

Funding *Arijit Ghosh*: Arijit Ghosh acknowledges partial support from the Science and Engineering Research Board (SERB), Government of India, through the MATRICS grant MTR/2023/001527, and from the Department of Science and Technology (DST), Government of India, through grant TPN-104427.

Acknowledgements We thank an anonymous reviewer for their suggestions in improving the query complexity from $\log^3 m$ (in an earlier version) to $\log^2 m$.



© Debarshi Chanda, Buddha Dev Das, Arijit Ghosh, and Gopinath Mishra;
licensed under Creative Commons License CC-BY 4.0



1 Introduction

We consider the problem of estimating the number of edges in an undirected, unweighted, simple graph using a sublinear number of queries. Let $G = (V, E)$ be a graph with $V = [n] := \{1, 2, \ldots, n\}$ vertices and $|E| = m$ edges. Given query access to G and an approximation parameter $\varepsilon \in (0, 1)$, the objective is to output an estimate $\widehat{m}$ such that

$$(1-\varepsilon)m \;\le\; \widehat{m} \;\le\; (1+\varepsilon)m$$

with success probability at least $2/3$.

Query model and our result

We work in a query model built on *Independent Set* (`IS`) queries, introduced by [6]. The `IS` query takes a subset of vertices $U \subseteq V$ and returns whether U is an independent set.

We consider the *Independent Set with Certificate* query, denoted by `ISC`, which takes as input a subset of vertices $U \subseteq V$ and returns

$$\texttt{ISC}(U) = \begin{cases} e \sim \mathrm{Unif}(E(U)), & \text{if } E(U) \neq \emptyset, \\ 0, & \text{otherwise,} \end{cases}$$

where $E(U)$ denotes the set of edges in the subgraph of G induced by U, and $\mathrm{Unif}(S)$ denotes the uniform distribution over a (nonempty) set S.

Importantly, the `ISC` query preserves the structural constraint of a standard `IS` query: the algorithm specifies exactly the same vertex subsets as in the IS model, and the only difference lies in the response, which returns a uniformly random edge as a certificate whenever the induced subgraph is nonempty.

[6] showed that the number of edges in a graph can be estimated using

$$\widetilde{O}\left(\min\left\{\frac{n^2}{m}, \sqrt{m}\right\}\right) \;=\; \widetilde{O}(n^{2/3})$$

`IS` queries, establishing a polynomial separation from local query based algorithms that require $\Theta(n)$ queries in the worst case [27]. In a subsequent work, [21] nearly resolved the query complexity of edge estimation using IS queries by showing that

$$\widetilde{\Theta}\left(\min\left\{\frac{n}{\sqrt{m}}, \sqrt{m}\right\}\right)$$

queries are both necessary and sufficient. In particular, this implies a worst-case complexity of $\widetilde{\Theta}(\sqrt{n})$ `IS` queries.

Our main result shows that the natural enrichment of `IS` queries with random certificates yields an exponential separation from standard `IS` queries for edge estimation, achieving polylogarithmic query complexity in m.

▶ **Theorem 1** (Main result). *There exists a randomized algorithm that, given access to a graph G through `ISC` queries and an approximation parameter $\varepsilon \in (0, 1/3)$, outputs an estimate $\widehat{m}$ of the number of edges m in G such that, with probability at least $2/3$,*

$$(1-\varepsilon)m \;\le\; \widehat{m} \;\le\; (1+\varepsilon)m.$$

The algorithm uses $O(\varepsilon^{-2} \log^2 m \cdot \log\log m)$ `ISC` queries in expectation.

The query complexity of the algorithm is *strictly output-sensitive* in the sense that it depends only on the number of edges, with no dependence on n, even in lower-order terms.

Theorem 1 not only demonstrates the power gained by augmenting IS access with random certificates over standard IS access, but also underscores the strength of sampling within induced subgraphs. When edges can only be sampled uniformly at random from the entire graph, estimating the number of edges requires $\widetilde{\Theta}(\sqrt{m})$ samples [30]. In contrast, allowing the algorithm to sample a uniformly random edge from a queried induced subgraph yields an exponential separation from global random edge-sampling, achieving query complexity $\widetilde{O}(\log^2 m)$.

In what follows, we highlight conceptual connections between our results and query model with several well-studied themes, including group testing, graph property testing, the power of random certificates, decision versus counting problems, and conditional sampling.

Group testing perspective

The introduction of IS queries in graph algorithms was originally motivated by ideas from *group testing*. In classical group testing, introduced by [25] and studied extensively since then, the goal is to infer the size of an unknown set of defective items by querying subsets of a known universe [3]. A standard group test returns only a binary result indicating whether the queried subset contains at least one defective item.

This viewpoint applies naturally to graph edge estimation. Here, edges play the role of defective items, and the allowed vertex pairs for queries impose structural constraints on the possible subsets of edges that can be queried. [6] introduced both IS and Bipartite Independent Set (BIS) queries through this lens. A BIS query takes two disjoint vertex sets as input and returns whether there exists an edge with one endpoint in each set.

In the same vein, [13] and [5] introduced the Edge Emptiness query EE to study triangle estimation and connectivity in graphs, respectively. An EE query specifies a set of candidate edges and reports whether at least one of them exists in the graph.

Decision versus counting

In a series of seminal works, [32, 33] studied the relationship between approximate counting and decision access for Boolean formulas. In particular, they showed that given decision access to satisfiability, one can approximately count the number of satisfying assignments of any Boolean formula on n variables using only $O(\log n)$ such queries.

Our work establishes an analogous phenomenon in the setting of graph edge estimation: augmenting decision-style IS queries with a random certificate enables efficient approximate counting of edges in a graph.

Conditional sampling

Our model also bears a close conceptual relationship to work on property testing and estimation with conditional sampling access. In distribution testing, a conditional sampling oracle [18, 16] allows the algorithm to specify a subset of the domain and receive a sample drawn from the distribution restricted to that subset. Such structured sampling can lead to exponential savings over standard sampling in tasks such as uniformity testing, support size estimation, and testing of joint distributions [15, 9, 14, 10].

In our setting, the ISC oracle plays an analogous role: the algorithm specifies a vertex subset, and if the induced subgraph is nonempty, it receives a uniformly random edge from that subgraph. Thus, ISC can be viewed as a structured conditional sampling primitive over

the edge set of the graph, where the conditioning is constrained to induced subgraphs. Our results demonstrate that, as in distribution testing, such conditional access can fundamentally change the complexity of estimation problems in graphs.

Power of a random certificate

Studying the power of random certificates for approximate counting is an important question across a variety of query models [31, 17]. These works show that augmenting group queries with access to random certificates can lead to significant reductions in query complexity.

In the graph setting, several queries have been introduced from the viewpoint of group queries while retaining the structural properties of the graph. Two notable examples are `IS` and `BIS`, introduced by [6] for estimating the number of edges. Our `ISC` query fits naturally within this framework: it preserves the structure of `IS` while adding a random certificate. In particular, our algorithm achieves substantially lower query complexity compared to algorithms using only `IS` access, and also improves over the best known $\widetilde{O}(\log^5 n)$-query algorithm based on `BIS` access due to [2].

The main difference between the power of random certificates studied in [31, 17] and our setting is that our query oracles are group graph queries that retain the structural properties of the underlying graph.

Graph property testing

Estimating the number of edges in a graph with sublinear queries has been studied in several access models. Early works focused on local queries such as degree, neighbour, and pair queries [28, 29, 27, 34], which provide fine-grained access but face inherent limits for edge estimation.

This led to models that incorporate global information. In particular, augmenting local access with random edge sampling [4, 26, 7] enables more efficient estimation. More recently, a finer-grained analysis of edge estimation under local queries enriched with random edge sampling have provided an understanding of the tradeoffs between local and global access [11, 12, 19, 7].

Another line of work studies structured global queries, notably `IS` and `BIS` [6, 21, 2]. These queries provide decision-only access over vertex subsets while preserving graph structure, and have been extended to hypergraphs [23, 8, 24]. We discussed their known upper and lower bounds earlier in the Introduction.

Finally, [1] showed that augmenting `IS` queries with additional local queries yields a quadratic separation from using `IS` queries alone, further illustrating the power of combining different query primitives. Other than the `BIS` query, all query models discussed above requires polynomial number of queries in the worst case.

Notations

Throughout, the notation $\widetilde{O}(f)$, suppress factors polynomial in $1/\varepsilon$ and $\log f$. For an integer k, we define $[k] = \{1, \ldots, k\}$. We write $a = (1 \pm \varepsilon)b$ to indicate that a is a $(1 \pm \varepsilon)$-approximation of b.

Throughout, we use $G = (V, E)$ to denote the input graph with $m = |E|$ number of edges and $V = |V|$ vertices. For a subset $U \subseteq V$, let $m(U)$ be the number of edges in the graph induced by the subset U.

2 Technical overview

We give a high-level description of our algorithm for estimating the number of edges in an undirected graph using Independent Set with Certificate (`ISC`) queries. Conceptually, our approach converts approximate counting into a sequence of controlled decisions and sampling steps that gradually reduce the graph while preserving a constant fraction of its edges.

A central conceptual contribution is the use of *approximation preserving subsets* and *approximation preserving sequences*, which formalise how much of the graph's edge mass is retained at each stage and guide both the algorithm and its analysis. We leverage graph sparsification techniques to enable iterative construction of approximation-preserving sequences on graphs using `ISC` queries.

Outline of the Algorithm

At a high level, our algorithm maintains a nested sequence of vertex subsets

$$V = U_0 \supseteq U_1 \supseteq \cdots \supseteq U_\kappa,$$

where each U_j contains a constant fraction of the edges of U_{j-1}. The algorithm alternates between two modes:

- If the current induced subgraph is "small" (that is, has fewer than τ edges), we estimate its edge count directly.
- Otherwise, we construct a smaller approximation preserving subset of the graph by randomly partitioning the vertex set and selecting a part that provably retains a constant fraction of the edges.

This iterative reduction of the graph continues until we reach a subset U_κ whose edge count is small enough to estimate accurately; we set the threshold for U_κ to be a sufficiently large constant. The final estimate of m is then reconstructed by multiplying the relative edge losses across all steps.

Where the $\widetilde{O}(\log^2 m)$ query complexity comes from

A key novelty of our approach is that all complexity bounds are driven by $\log m$ rather than $\log n$. This is achieved through three ideas that are not present in prior `IS` or `BIS` based algorithms:

- *Random certificates enable output-sensitive sparsification:* Unlike decision-only `IS` or `BIS` queries, ISC queries allow us to estimate how many edges of a subset lie in a randomly chosen part, which lets us iteratively construct smaller induced subgraphs to work on while tracking edge mass locally.
- *Approximation Preserving Sequence:* Instead of estimating m directly, we construct a chain of subsets $U_0 \supseteq U_1 \supseteq \cdots \supseteq U_\kappa$ such that each U_j retains a constant fraction of edges of U_{j-1}. Crucially, we establish κ to be $O(\log m)$ in expectation, independent of n. We also bound the number of queries required to construct this sequence as $O(\kappa^2)$ and bound its expectation.
- *Local multiplicative reconstruction:* We only need to estimate the ratios $p_j = \frac{m(U_j)}{m(U_{j-1})}$, each of which is $\Omega(1)$ due to the $U_0, \ldots, U_\kappa$ being an approximation preserving subsequence. This keeps the per-level query complexity small and makes the overall cost in expectation $\widetilde{O}(\kappa^2) = \widetilde{O}(\log^2 m)$.

Estimating Small Edge Counts

Fix a threshold τ. One can design an algorithm that makes $O(\sqrt{\tau})$ `ISC` queries and guarantees the following with desired probability about a given subset $U \subseteq V$. If $m(U) \leq \tau$, then the algorithm reports an approximation of $m(U)$. Moreover, if $m(U) \geq 8\tau$, the algorithm reports that $m(U)$ is large. The proof is based on the birthday paradox. The details can be found in Section 3.

Graph Sparsification

The idea here is to leverage graph sparsification to construct a smaller subset of vertices to work on at each step. We partition U uniformly at random into two subsets U^1 and U^2. A graph sparsification lemma, along the result of [6], shows that with desired probability,

$$\left| \frac{m(U)}{2} - \sum_{i=1}^{2} m(U^i) \right| \leq c\sqrt{m(U)} \log m(U).$$

We provide the proof in Section 4. Intuitively, this means that the total number of edges inside U^1 and U^2 is close to half of $m(U)$, up to a lower-order error term. As long as $m(U)$ is sufficiently large, this implies that at least one of U^1 or U^2 must contain a constant fraction of the edges of U. More precisely, there exists an absolute constant ζ_{sp} such that if $m(U) \geq \zeta_{\text{sp}}$, then with probability at least $1 - 1/m(U)$, at least one of $m(U^1)/m(U)$ and $m(U^2)/m(U)$ is at least $1/8$, and both $m(U^1)/m(U)$ and $m(U^2)/m(U)$ are at most $3/4$.

Choosing the Good Part

After partitioning U into U^1 and U^2, we must identify which part retains an appropriate fraction of the edges. Note that this step is fundamentally enabled by the `ISC` queries. The idea is to take a random sample of the edges in U, and check whether it is preserved in the induced subgraph of either U^1 or U^2.

To do this, we draw random edges from U via `ISC` queries. Let $\widehat{p}(U^i)$ be the fraction of sampled edges that fall inside U^i, for $i \in [2]$. If $\widehat{p}(U^i) \geq 1/16$, we select U^i as the next subset; otherwise, we pick an arbitrary part. A Chernoff bound guarantees that, with desired probability, we correctly identify a subset satisfying

$$\frac{1}{32} \leq \frac{m(U^i)}{m(U)} \leq \frac{3}{4}.$$

Such subsets are called *approximation preserving subsets*. The upper bound of $3/4$ is useful to bound the final query complexity and the lower bound of $1/32$ is useful for guaranteeing the quality of the estimator.

Constructing an Approximation Preserving Sequence

Starting with $U_0 = V$, we iterate:

1. Run threshold checking on U_j.
2. If $m(U_j) < \tau$, stop and set $U_\kappa = U_j$.
3. Otherwise, sparsify U_j and select an approximation preserving subset U_{j+1}.

This yields a sequence $V = U_0 \supseteq U_1 \supseteq \cdots \supseteq U_\kappa$, where, with probability at least $4/5$, each U_j satisfies

$$\frac{1}{32} \leq \frac{m(U_j)}{m(U_{j-1})} \leq \frac{3}{4}.$$

Final Estimation

For each $j \in [\kappa]$, define

$$p_j = \frac{m(U_j)}{m(U_{j-1})}.$$

Then we have the identity

$$m = \frac{m(U_\kappa)}{\prod_{j=1}^{\kappa} p_j}.$$

Instead of estimating each $m(U_j)$ from scratch, we estimate only the ratios p_j. Because each $p_j = \Omega(1)$ by construction of the approximation preserving sequence, we can estimate each p_j using small number of queries, independent of n or $m(U_j)$. This yields an estimate $\widehat{p}_j$ satisfying the following with desired probability.

$$\widehat{p}_j = (1 \pm \varepsilon/O(\sqrt{\kappa}))\, p_j$$

Note that the entire approximation preserving subsequence is constructed a priori. Hence, we know κ during the final estimation phase that we use to fix the approximation parameters appropriately for the multiplicative estimation for each $p_j, j \in [\kappa]$. Let $\widehat{m}(U_\kappa)$ be the estimate obtained via the birthday-based estimator. Our final output is

$$\widehat{m} = \frac{\widehat{m}(U_\kappa)}{\prod_{j=1}^{\kappa} \widehat{p}_j}.$$

A careful multiplicative error analysis shows that this is a $(1 \pm \varepsilon)$-approximation to m.

Proof Roadmap

The correctness proof proceeds in the following stages:

- *Threshold correctness:* We show that the collision-based test reliably distinguishes between $m(U) < \tau$ and $m(U) \geq 8\tau$, and that the birthday-paradox estimator is accurate when $m(U) < \tau$.
- *Constructing approximation preserving subsequence:* While we decrease the failure probability of choosing a good subset at each iteration, the probability of failure of the sparsification step depends on the number of edges remaining. This necessitates a careful analysis of the failure probability of the sparsification step and of our threshold-checking mechanisms to establish the correctness of our algorithm and query complexity bounds.
- *Multiplicative reconstruction:* Given our approximation preserving subsequence, we estimate each p_j with appropriately scaled approximation error. Then, we bound the cumulative error from estimating all p_j and show that the product estimator preserves a global $(1 \pm \varepsilon)$ approximation.
- *Complexity analysis:* We show that $\mathbb{E}[\kappa] = O(\log m)$ and that the total query complexity is dominated by estimating the p_j's, yielding an overall expected complexity of $\widetilde{O}(\log^2 m/\varepsilon^2)$.

Outline

We begin in Section 3, where we present the threshold-checking algorithm and show how to estimate the edge count when the number of edges is below a given threshold. In Section 4, we describe the algorithm for constructing an approximation-preserving subsequence. In Section 4.1, we establish how graph sparsification yields an approximation-preserving subset,

and use this result in Section 4.2 to construct an approximation-preserving subsequence. Finally, in Section 5, we present the complete algorithm for edge estimation. Appendix A discusses some useful concentration inequalities.

3 Threshold checking

Let a fixed threshold ($\tau \geq 1$) be given. We consider the following threshold-checking problem: given a subset of vertices $U \subseteq V$, determine whether $m(U) < \tau$. If this condition holds, the goal is to estimate the $(1 \pm \varepsilon)$ approximation of $m(U)$.

To address this task, we present two algorithms `IsCollision` and `BirthdayEstimate` based on the birthday paradox techniques [30, Chapter 5]. `IsCollision` distinguishes between the cases $m(U) < \tau$ or $m(U) \geq 8\tau$ (Lemma 2). `BirthdayEstimate` outputs an $(1 \pm \varepsilon)$ approximation of $m(U)$, if $m(U) < \tau$ (Lemma 3). The corresponding guarantees are stated below. For details, see the full version of the paper [20].

▶ **Lemma 2** (Correctness of `IsCollision`). *For a subset $U \subseteq V$ and a threshold $\tau \geq 1$, the following statements hold for `IsCollision` with probability at least $1 - \delta$:*

(i) *If $m(U) \leq \tau$, then `IsCollision` outputs 1.*

(ii) *If $m(U) \geq 8\tau$, then `IsCollision` outputs 0.*

The number of `ISC` queries made is $O(\sqrt{\tau}\log(\delta^{-1}))$

Suppose that a collision is detected, i.e., `IsCollision` outputs 1. We can obtain a $(1 \pm \varepsilon)$-approximation of $m(U)$ using additional `ISC` queries. This estimation procedure is implemented in `BirthdayEstimate`.

▶ **Lemma 3** (Correctness of `BirthdayEstimate`). *For a subset $U \subseteq V$, threshold $\tau \geq 1$ and approximation error $\varepsilon > 0$. If $m(U) \leq \tau$, then `BirthdayEstimate` outputs an $(1 \pm \varepsilon)$ estimate of $m(U)$ with probability at least $15/16$. The number of `ISC` query made is $\Theta\left(\varepsilon^{-1}\sqrt{\tau}\right)$.*

4 Sparsification

In this section, we will describe a procedure to generate an approximation preserving sequence (formally defined below) $U_0, \ldots, U_\kappa$ with probability at least $7/8$. Moreover, the expected query complexity of the procedure is $O(\log^2 m)$.

▶ **Definition 4** (Approximation preserving subset). *For $U, U' \subset V$ such that $U' \subseteq U$, we say that U' is an* approximation preserving subset *of U if*

$$\frac{1}{32} \leq \frac{m(U')}{m(U)} \leq \frac{3}{4}.$$

▶ **Definition 5** (Approximation preserving sequence). *For a sequence $U_0, U_1, \ldots, U_\kappa \subseteq V$ with $U_0 \supseteq U_1 \supseteq \ldots \supseteq U_\kappa$, We say that $U_0, U_1, \ldots, U_\kappa$ is an* approximation preserving sequence *if U_{i-1} is an approximation preserving subset of U_i for all $i \in [\kappa]$.*

Formally, we will prove the following lemma in this section.

▶ **Lemma 6** (Approximate sequence lemma). *There exists an algorithm that outputs an approximation preserving sequence $U_0, U_1, \ldots, U_\kappa$ with probability at least $4/5$. The expected value of κ is $O(\log m)$, and the expected number of `ISC` queries made is $O(\log^2 m)$.*

To prove the above lemma, we start with $U_0 = V$, and find U_i iteratively. Informally speaking, we stop with U_κ when $m(U_\kappa)$ is *small enough*. In Section 4.1, we discuss a procedure (`ApproxSubset` corresponding to Lemma 7) that generates a U' which is an approximation preserving subset of U, with specified probability, for any given U as long as $m(U)$ is *large*. In Section 4.2, we describe the procedure (`ApproxSequence` corresponding to Lemma 12) that uses `IsCollision` and `ApproxSubset` iteratively to generate an approximation preserving subsequence $U_0, \ldots, U_\kappa$. Essentially, `IsCollision` is used to determine whether the number of edges in the current subset U_i is small or large and `ApproxSubset` is used to generate a good subset as long as $m(U_i)$ is large.

4.1 Procedure to find approximate preserving subset

In this section, we describe the procedure `ApproxSubset` (see Algorithm 1) for finding the approximation preserving subset $U' \subseteq U$, for any given U and prove its guarantee.

Algorithm 1 `ApproxSubset`(U, δ).

Require: `ISC` query access to G and a confidence parameter $\delta \in (0, 1)$.
Ensure: A subset $U' \subseteq U$.

```
Uniformly and independently partition U into U^1 and U^2.
Uniformly and independently sample O(log(δ^{-1})) random edges from U and let S be
    the multi-set of sampled edges.
Compute p̂(U^i) = (1/|S|) Σ_{e∈S} 1_{e∈U^i} for i ∈ {1,2}.
if There exists i ∈ {1,2} such that p̂(U^i) ≥ 1/16 then
    return U^i as U'
else
    return U^1 or U^2 arbitrarily as U'.
end if
```

Formally, we will prove the following.

▶ **Lemma 7** (Approximate subset lemma). *Let ζ_{sp} be a suitably large constant. Consider `ApproxSubset` that takes $U \subseteq V$ as input and produces $U' \subseteq U$ as the output.*

- **(i)** $\mathbb{E}[m(U')] \leq m(U)/2$;
- **(ii)** *If $m(U) \geq \zeta_{\text{sp}}$, with probability $1 - \delta - 1/m(U)$, U' is an approximation preserving subset of U.*

The construction of an approximation-preserving subset U' proceeds in two steps. First, we sparsify the graph by randomly partitioning the set U into U^1 and U^2 (see Lemma 8). When $m(U)$ is large enough, this partition guarantees the existence of U^i such that $m(U^i)$ is a constant fraction of $m(U)$ (Corollary 10). In the second step, we identify such an approximation-preserving subset from U^1 and U^2 (Lemma 11). Combining these results yields the proof of Lemma 7.

We prove the following variant of the graph sparsification result of Beame et al. [6].

▶ **Lemma 8** (Graph Sparsification). *For a given $\delta \in (0, 1)$, there exists an absolute constant c such that the following holds. Let $G = (V, E)$ be a graph with n vertices and m edges. Let U^1 and U^2 be a uniformly random partition of V. Then,*

- **(i)** $\mathbb{E}[m(U^1) + m(U^2)] = m(U)/2$;
- **(ii)** $\Pr\left[\left|m/2 - \sum_{i=1}^{2} m(U^i)\right| \geq c\sqrt{m}\log(m/\delta)\right] \leq \delta$.

We will need the following lemma to prove the above graph sparsification lemma.

▶ **Lemma 9.** *Let A be a set of ℓ elements. For every element $t \in A$, one assigns a color independently and uniformly from $\{1,2\}$. Let $C^{(i)}$ be the number of vertices in A with color i, for any $i \in \{1,2\}$. Then for $\delta \in (0,1)$, we have*

$$\Pr\left[\left|C^{(1)} - C^{(2)}\right| \geq \sqrt{6\ell \log(4\delta^{-1})}\right] \leq \delta \text{ and } \mathbb{E}\left[\left|C^{(1)} - C^{(2)}\right|\right] \leq \sqrt{\ell}$$

Proof. For $t \in A$, let X_t be the indicator random variable that is 1 if t is assigned color 1 and 0 otherwise. Then, $C^{(1)} = \sum_{t \in A} X_t$ and $\mathbb{E}\left[C^{(1)}\right] = \ell/2$. Using Chernoff's inequality (Lemma 19) with $\varepsilon = \sqrt{(6/\ell)\log(4/\delta)}$, we have

$$\Pr\left[\left|C^{(1)} - \frac{\ell}{2}\right| > \frac{\varepsilon\ell}{2}\right] \leq 2\exp\left(\frac{6\log(4\delta^{-1})}{\ell} \cdot \frac{\ell}{6}\right) \leq \frac{\delta}{2} \tag{1}$$

Similarly, we can have the same bound for color 2.

$$\Pr\left[\left|C^{(2)} - \frac{\ell}{2}\right| > \frac{\varepsilon\ell}{2}\right] \leq \frac{\delta}{2} \tag{2}$$

Observe that $\left|C^{(1)} - C^{(2)}\right| \leq \left|C^{(1)} - (\varepsilon\ell/2)\right| + \left|C^{(2)} - (\varepsilon\ell/2)\right|$. Therefore

$$\Pr\left[\left|C^{(1)} - C^{(2)}\right| > \varepsilon\ell\right] \leq \delta$$

For $t \in A$, let Y_t be a random variable that is 1 if t is assigned the color 1 and -1 otherwise. Then $C^{(1)} - C^{(2)} = \sum_{t \in A} Y_t = Y$. We have that $\Pr[Y_t = 1] = \Pr[Y_t = 1] = 1/2$, then

$$\mathbb{E}[Y_t] = 0 \text{ and } \mathbb{E}[Y_t^2] = 1$$

By the independence of Y_t, we get

$$\mathbb{E}[Y^2] = \sum_{t \in A} \mathbb{E}[Y_t^2] + \sum_{u \neq v} \mathbb{E}[Y_u Y_v] = \ell$$

Finally, we have $\mathbb{E}[|Y|] \leq \sqrt{\mathbb{E}[|Y|^2]} = \sqrt{\mathbb{E}[Y^2]} \leq \sqrt{\ell}$. ◀

We now prove Lemma 8.

Proof of Lemma 8. Let each vertex $t \in V$ be associated with a random variable $Y_t \in \{1,2\}$ such that $\Pr[Y_t = 1] = 1/2$. For $i \in \{1,2\}$, define $U^i = \{t \in V \mid Y_t = i\}$ and

$$f(Y_1, \ldots, Y_n) = m(U^1) + m(U^2).$$

A fixed edge is counted by f if and only if both its endpoints are assigned the same label, which occurs with probability $1/2$. Therefore, $\mathbb{E}[f] = m/2$.

Consider the Doob martingale $X_0, X_1, \ldots, X_n$, where $X_t = \mathbb{E}[f(Y_1, \ldots, Y_n) | Y_1, \ldots, Y_t]$. For a fix values of $Y_1, \ldots, Y_{t-1}$, we have,

$$X_{t-1} = \frac{g(1) + g(2)}{2}, \text{ where } g(i) = \mathbb{E}[f(Y_1, \ldots, Y_n) | (Y_1, \ldots, Y_{t-1}) \cap (Y_t = i)]$$

Since, X_t is either $g(1)$ or $g(2)$, we have $|X_t - X_{t-1}| \leq |g(1) - g(2)|$. Let $N(t)$ be the set of neighbours of the vertex t and $\deg(t) = |N(t)|$. Let $N_{<t} = N(t) \cap [t-1]$ and

$N_{>t} = N(t) \cap \{(t+1), \ldots, n\}$. Let $C^{(i)}_{<t}$ and $C^{(i)}_{>t}$ be the number of vertices in $N_{<t}$ and $N_{>t}$ respectively. So,

$$\Delta_t = |g(1) - g(2)| = \left| C^{(1)}_{<t} + \mathbb{E}\left[C^{(1)}_{>t} \mid Y_t = 1\right] - C^{(2)}_{<t} - \mathbb{E}\left[C^{(2)}_{>t} \mid Y_t = 2\right] \right|$$

The above equation is true because any edge with vertices in $[t-1]$ will have the same contribution to $g(1)$ and $g(2)$. Similarly, any edge which have one vertex in $[t-1]$ and another in $\{(t+1), \ldots, n\}$ or both of its vertices in $\{(t+1), \ldots, n\}$ will have the same contribution. Using Lemma 9, we have with probability at least $1 - \delta'$,

$$\Delta_t \leq \sqrt{6\deg(t) \log \frac{4}{\delta'}} + \sqrt{\deg(t)} \leq 2c\sqrt{\deg(t) \log \frac{1}{\delta'}} = c_t$$

Let $\mathcal{B}$ be the event that there exits a t such that $\Delta_t > c_t$. Let n_a be the set of active vertices (vertices with degree at least 1). Therefore, using union bound $\Pr[\mathcal{B}] \leq 4n_a\delta' \leq 4m\delta'$.

Let $S = \sum_{t=1}^{n} c_t^2 = \sum_{t=1}^{n} 4c^2 \deg(t) \log(4/\delta') = O(m \log(1/\delta'))$ and $s = c\sqrt{m}\log(1/\delta')$. Using Lemma 21, we have

$$\Pr\left[\left|f - \frac{m}{2}\right| > s\right] \leq 2\exp\left(-\frac{s^2}{2S}\right) + \Pr[\mathcal{B}] \leq \delta' + 4m\delta'$$

Replacing δ' with δ/cm for a sufficiently large constant, we obtain

$$\Pr\left[\left|\frac{m}{2} - \sum_{i=1}^{2} m(U^i)\right| \geq c\sqrt{m}\log\left(\frac{m}{\delta}\right)\right] \leq \delta$$

which completes the proof. ◀

▶ **Corollary 10.** *For a subset $U \subseteq V$, suppose there exists a sufficiently large absolute constant ζ_{sp} such that $m(U) \geq \zeta_{\text{sp}}$. Let U^1 and U^2 be a uniformly random partition of U. Then, with probability at least $1 - \frac{1}{m(U)}$, the following hold.*

- **(i)** $\frac{m(U^1)}{m(U)} \geq 1/8$ *or* $\frac{m(U^2)}{m(U)} \geq 1/8$;
- **(ii)** $\frac{m(U^1)}{m(U)} \leq 3/4$ *and* $\frac{m(U^2)}{m(U)} \leq 3/4$.

Proof. Taking ζ_{sp} to be sufficiently large, we have $c\sqrt{m(U)}\log m(U) \leq \frac{m(U)}{4}$. So, by taking $\delta = 1/m(U)$ in Lemma 8, we

$$\Pr\left[\frac{m(U)}{4} \leq \sum_{i=1}^{2} m(U^i) \leq \frac{3\,m(U)}{4}\right] \geq 1 - \frac{1}{m(U)}\,.$$

This implies that, with probability at least $1 - 1/m(U)$, at least one of $m(U^1)/m(U)$ and $m(U^2)/m(U)$ is at least $1/8$, and both $m(U^1)/m(U)$ and $m(U^2)/m(U)$ are at most $3/4$. ◀

To prove the guarantees of `ApproxSubset` described in Algorithm 1, we need Lemma 11.

▶ **Lemma 11.** *With probability at least $1-\delta$, the following statements hold for `ApproxSubset`*

- **(i)** *If $\frac{m(U^i)}{m(U)} \geq \frac{1}{8}$, then $\widehat{p}(U^i) \geq \frac{1}{16}$.*
- **(ii)** *If $\frac{m(U^i)}{m(U)} < \frac{1}{32}$, then $\widehat{p}(U^i) < \frac{1}{16}$.*

The number of `ISC` queries made is $O\left(\log(\delta^{-1})\right)$.

Proof. Consider a subset $U^i \subseteq U$ for any $i \in \{1,2\}$. For each sampled edge, define the indicator random variable

$$X_j = \begin{cases} 1 & \text{if the } j^{\text{th}} \text{ sampled edge lies in } U^i, \\ 0 & \text{otherwise.} \end{cases}$$

Observe that $\mathbb{E}[X_j] = \frac{m(U^i)}{m(U)}$.

Define the fraction of sampled edges that belong to U^i by $X = \frac{1}{|S|}\sum_{j\in[|S|]} X_j$. where S is the multiset of sampled edges (Line 2) obtained via `ISC` queries on U. Note that $X = \widehat{p}(U^i)$ and hence

$$\mathbb{E}[\widehat{p}(U^i)] = \frac{1}{|S|}\sum_{j\in[|S|]} \mathbb{E}[X_j] = \frac{m(U^i)}{m(U)} = \frac{m(U^i)}{m(U)}.$$

Applying Hoeffding's bound (Lemma 20), we get

$$\Pr\left(\left|\widehat{p}(U^i) - \frac{m(U^i)}{m(U)}\right| \geq \tfrac{1}{32}\right) \leq \exp\left(-\frac{|S|}{2}\cdot(1/32)^2\right) \leq \delta/2.$$

The last inequation holds since $|S| = O(\log 1/\delta)$. Using the union bound over $i \in \{1,2\}$, we get the desired result. ◀

We now give the proof of Lemma 7, using Lemma 8, Corollary 10 and Lemma 11

Proof of Lemma 7. In `ApproxSubset`, the set U is partitioned uniformly at random into U^1 and U^2. From Lemma 8(i),

$$\mathbb{E}[m(U^1) + m(U^2)] = m(U)/2.$$

Since U' is either U^1 or U^2, it follows that $\mathbb{E}[m(U')] \leq m(U)/2$, proving part (i).

We now prove part (ii). Let $\mathcal{E}_{\text{apx}}$ denote the event that U' is an approximation-preserving subset of U, that is, $1/32 \leq m(U')/m(U) \leq 3/4$. Let $\mathcal{E}$ be the event that at least one of $m(U^1)/m(U)$ and $m(U^2)/m(U)$ is at least $1/8$, and both $m(U^1)/m(U)$ and $m(U^2)/m(U)$ are at most $3/4$. By Corollary 10,

$$\Pr[\mathcal{E}] \geq 1 - \frac{1}{m(U)}.$$

Let us condition on the event $\mathcal{E}$. It suffices to bound $\Pr[\mathcal{E}_{\text{apx}} \mid \mathcal{E}]$. Then both $m(U^1)/m(U)$ and $m(U^2)/m(U)$ are at most $3/4$, and at least one of $m(U^1)/m(U)$ and $m(U^2)/m(U)$ is at least $1/8$; w.l.o.g. assume $m(U^1)/m(U) \geq 1/8$. If $m(U^2)/m(U) \geq 1/32$, then since $U' \in \{U^1, U^2\}$ we immediately have $1/32 \leq m(U')/m(U) \leq 3/4$, that is, $\Pr[\mathcal{E}_{\text{apx}} \mid \mathcal{E}] = 1$. Otherwise $m(U^2)/m(U) < 1/32$. By Lemma 11, with probability at least $1-\delta$, we obtain $\widehat{p}(U^1) \geq 1/16$ and $\widehat{p}(U^2) < 1/16$, implying $U' = U^1$. Consequently $1/32 \leq m(U')/m(U) \leq 3/4$ holds with probability at least $1-\delta$ conditioned on $\mathcal{E}$. Thus in all cases

$$\Pr[\mathcal{E}_{\text{apx}} \mid \mathcal{E}] \geq 1 - \delta.$$

Therefore,

$$\Pr[\mathcal{E}_{\text{apx}}] \geq \Pr[\mathcal{E}] \cdot \Pr[\mathcal{E}_{\text{apx}} \mid \mathcal{E}] \geq 1 - \frac{1}{m(U)} - \delta.$$ ◀

Algorithm 2 `ApproxSequence`(V).

Require: `ISC` access to graph G.
Ensure: A nested sequence $U_1, U_2, \cdots, U_\kappa$.
1: Initialize $U_0 \leftarrow V$; *coll* $\leftarrow 1$; $\kappa \leftarrow 0$
2: **while** *true* **do**
3: *coll* $\leftarrow$ `IsCollision`$(U_\kappa, \zeta_{\text{sp}}, \frac{1}{2^{\kappa+5}})$
4: **if** *coll* equals 1 **then**
5: **break**
6: **else**
7: $U_{\kappa+1} \leftarrow$ `ApproxSubset`$(U_\kappa, \frac{1}{2^{\kappa+5}})$
8: $\kappa \leftarrow \kappa + 1$
9: **end if**
10: **end while**
11: **return** $U_1, U_2, \cdots, U_\kappa$

4.2 Procedure to find approximate preserving sequence

We now describe the algorithm `ApproxSequence` (see Algorithm 2), which constructs the sparsification sequence used by the main estimator.

▶ **Lemma 12** (Correctness of `ApproxSequence`). *Consider* ***`ApproxSequence`*** *as described in Algorithm 2. With probability at least* $4/5$*, it outputs an approximation preserving sequence* $U_1, U_2, \ldots, U_\kappa$ *such that* $m(U_\kappa) \leq 8\zeta_{\text{sp}}$*. Here* ζ_{sp} *is given large constant. The number of* ***`ISC`*** *queries made is* $O(\kappa^2)$*.*

Proof.

Correctness. Let us consider the event $\mathcal{E}_{\text{col}}$ that denotes

- There does not exist a j such that the algorithm goes to the next iteration if $m(U_j) \leq \zeta_{\text{sp}}$.
- For any j such that the algorithm goes to the next iteration if $m(U_j) \geq 8\zeta_{\text{sp}}$.

The probability of $\mathcal{E}_{\text{col}}$ is lower bounded by `IsCollision` succeeds in all iterations during the run of the algorithm. Using the union bound, we have

$$\Pr[\mathcal{E}_{\text{col}}] \geq 1 - \sum_{i=1}^{\infty} \frac{1}{2^{i+5}} \geq 1 - \frac{1}{2^5} \sum_{i=1}^{\infty} \frac{1}{2^i} \geq 1 - \frac{1}{16} = \frac{15}{16}.$$

Let us continue the argument under the conditional space that $\mathcal{E}_{\text{col}}$ has occurred.

We assume $m(U_0) = m$ is sufficiently large; hence, the algorithm does not terminate in the first iteration, given that $\mathcal{E}_{\text{col}}$ has occurred. Let $U_0, \ldots, U_\kappa$ be the output of the algorithm, where κ is a positive integer. Conditioned on the event $\mathcal{E}_{\text{col}}$, we have $m(U_j) \geq \zeta_{\text{sp}}$ for every $j \leq \kappa - 1$.

We prove the claim by induction on κ. For each $j \geq 0$, let $\mathcal{E}_j$ denote the event which is that the algorithm executes iteration j and either terminates with, $m(U_j) \leq 8\zeta_{\text{sp}}$, or it produces an approximation preserving subset U_{j+1} such that $m(U_{j+1}) \leq \frac{3m(U_j)}{4}$.

Let $\mathcal{E}_{\leq j}$ denote the event that $\mathcal{E}_i$ occurs for all $i \in [j]$, that is,

$$\mathcal{E}_{\leq j} = \bigcap_{i=1}^{j} \mathcal{E}_i.$$

Equivalently, $\mathcal{E}_{\leq j}$ is the event that the algorithm runs for j iterations, such that $U_0, \ldots, U_j$ is an approximation preserving sequence, and at iteration j either it produces U_{j+1}, an approximation preserving subset of U_j such that $m(U_{j+1}) \leq \frac{3m(U_j)}{4}$, or the algorithm terminates with $m(U_j) \leq 8\zeta_{\text{sp}}$.

The induction hypothesis is that

$$\Pr[\mathcal{E}_{\leq j}] \;\geq\; 1 - \delta_{\leq j},$$

where

$$\delta_i \;=\; \frac{1}{m(U_i)} + \frac{1}{2^{i+5}}, \qquad \text{and} \qquad \delta_{\leq j} \;=\; \sum_{i=0}^{j} \delta_i.$$

We first prove the base case. That is, $\Pr[\mathcal{E}_0] \geq 1 - \delta_0$. Recall that we assume that $m(U_0) = m$ is sufficiently large. Moreover, we assume that event $\mathcal{E}_{\text{col}}$ has occurred and algorithm produced a sequence $U_0, \ldots, U_\kappa$ with $\kappa \geq 1$. So, $\mathcal{E}_0$ occurs if `ApproxSubset` reports an approximate preserving subset U_{j+1}. By Lemma 7, this occurs with probability at least $1 - \delta_0$.

For the inductive step, we mainly show that $\Pr[\mathcal{E}_j \mid \mathcal{E}_{\leq j-1}] \geq 1 - \delta_j$ for $j \geq 1$. Let us argue that it is sufficient. Note that $\delta_{\leq j} = \delta_{\leq j-1} + \delta_j$ for $j \geq 1$ and

$$\Pr[\mathcal{E}_{\leq j}] = \Pr[\mathcal{E}_{\leq j-1} \cap \mathcal{E}_j] = \Pr[\mathcal{E}_{\leq j-1}] \cdot \Pr[\mathcal{E}_j \mid \mathcal{E}_{\leq j-1}].$$

By the induction hypothesis, $\Pr[\mathcal{E}_{\leq j-1}] \geq 1 - \delta_{\leq j-1}$. We will show that

$$\Pr[\mathcal{E}_j \mid \mathcal{E}_{\leq j-1}] \geq 1 - \delta_j.$$

Therefore,

$$\begin{aligned}
\Pr\left[\mathcal{E}_{\leq j}\right] &= \Pr\left[\mathcal{E}_{\leq j-1}\right] \cdot \Pr\left[\mathcal{E}_j \mid \mathcal{E}_{\leq j-1}\right] \\
&\geq (1 - \delta_{\leq j-1})(1 - \delta_j) \\
&\geq 1 - \delta_{\leq j-1} - \delta_j \\
&= 1 - \delta_{\leq j}.
\end{aligned}$$

Consider any j with $1 \leq j \leq \kappa - 1$. Under the conditional space $\mathcal{E}_{\text{col}}$ and along with the fact that we assume $m(U_0) = m$ is sufficiently large, $m(U_j) \geq \zeta_{\text{sp}}$ for each $j \leq \kappa - 1$.

We consider two cases: $m(U_j) \geq 8\zeta_{\text{sp}}$, and $\zeta_{\text{sp}} \leq m(U_j) \leq 8\zeta_{\text{sp}}$, and analyse them individually.

Case 1: $(m(U_j) > 8\zeta_{\text{sp}})$. Recall that we are assuming that the event $\mathcal{E}_{\text{col}}$ has occurred. So, `ApproxSequence` does not terminate after executing `IsCollision` and execute `ApproxSubset`. Moreover it produces an approximate preserving subset U_{j+1} with $m(U_{j+1}) \leq \frac{3m(U_j)}{4}$, that is event $\mathcal{E}_j$ occurs, when `ApproxSubset` reports an approximate preserving subset U_{j+1} with $m(U_{j+1}) \leq \frac{3m(U_j)}{4}$. By Lemma 7 and Line 7 of `ApproxSequence`, this occurs with probability at least $1 - \delta_j$ where,

$$\delta_j = \frac{1}{m(U_j)} + \frac{1}{2^{j+5}}$$

Hence,

$$\Pr[\mathcal{E}_j \mid \mathcal{E}_{\leq j-1}] \geq 1 - \delta_j^{(1)},$$

Case 2: $(\zeta_{\text{sp}} \leq m(U_j) \leq 8\zeta_{\text{sp}})$. In this case, we show that

$$\Pr[\mathcal{E}_j \mid \mathcal{E}_{\leq j-1}] \geq 1 - \delta_j$$

regardless of whether `ApproxSequence` terminates after executing `IsCollision` or proceeds to execute `ApproxSubset`.

In the former case, the claim follows directly from the definition of $\mathcal{E}_j$ together with the fact that $m(U_j) \leq 8\zeta_{\text{sp}}$. In the latter case, `ApproxSequence` produces an approximate preserving subset U_{j+1} of U_j provided that `ApproxSubset` returns an approximate preserving subset U_{j+1}. By Lemma 7, this occurs with probability at least $1 - \delta_j$, where

$$\delta_j \leq \frac{1}{m(U_j)} + \frac{1}{2^{j+5}}.$$

Hence,

$$\Pr[\mathcal{E}_j \mid \mathcal{E}_{\leq j-1}] \geq 1 - \delta_j.$$

We are done with the proof of the inductive step. We summarize the above discussion as follows. $\Pr[\mathcal{E}_{\text{col}}] \geq 15/16$. When the event $\mathcal{E}_{\text{col}}$ occurs, $\Pr[\mathcal{E}_j \mid \mathcal{E}_{j-1}] \geq 1 - \delta_j$ for all $j \geq 0$. Moreover, $\Pr[\mathcal{E}_{\leq j}] \geq 1 - \delta_{\leq j}$ for all $j \geq 0$. Recall that the algorithm outputs a sequence $U_0, \ldots, U_{\kappa-1}$. The events $\mathcal{E}_{\text{col}}$ and $\mathcal{E}_{\leq \kappa-1}$ occur with probability at least

$$\Pr[\mathcal{E}_{\text{col}}] - \delta_{\leq \kappa-1}.$$

Hence, the following facts hold with the same probability:

1. $U_0, \ldots, U_\kappa$ is an approximate preserving sequence, that is,

$$\frac{1}{32} \leq \frac{m(U_i)}{m(U_{i-1})} \leq \frac{3}{4}, \text{ where } i \in [\kappa].$$

2. $m(U_j) \geq \zeta_{\text{sp}}$ for each $j \leq \kappa - 1$, and $m(U_\kappa) \leq 8\zeta_{\text{sp}}$.

Note that `ApproxSequence` produces an approximation-preserving sequence $U_1, U_2, \cdots, U_\kappa$ such that $m(U_\kappa) \leq 8\zeta_{\text{sp}}$ whenever both of the above conditions hold. This occurs with probability at least $1/5$, provided $\delta_{\kappa-1} \leq 1/10$, since $\Pr[\mathcal{E}_{\text{col}}] \geq 15/16$.

We complete the proof of correctness of `ApproxSequence` by showing that $\delta_{\kappa-1} \leq 1/10$.

$$\delta_{\leq \kappa-1} = \sum_{i=0}^{\kappa-1} \frac{1}{m(U_j)} + \frac{1}{2^{j+5}} \leq \sum_{i=0}^{\infty} \frac{1}{2^{j+4}} + \sum_{i=0}^{\kappa-1} \frac{1}{m(U_j)} \leq \frac{1}{16} + \sum_{i=0}^{\kappa-1} \frac{3^{\kappa-1-i}}{4^{\kappa-1-i}\zeta_{\text{sp}}}.$$

The last inequation holds since $m(U_{\kappa-1}) \geq \zeta_{\text{sp}}$ and $m(U_i) \leq \frac{3m(U_{i-1})}{4}$ for all $i \in [\kappa - 1]$. We obtain the bound of $\delta_{\leq \kappa-1}$ as below:

$$\delta_{\leq \kappa-1} = \frac{1}{16} + \frac{1}{\zeta_{\text{sp}}} \sum_{i=0}^{\kappa-1} \frac{3^{\kappa-1-i}}{4^{\kappa-1-i}} = \frac{1}{16} + \frac{1}{\zeta_{\text{sp}}} \sum_{i=0}^{\kappa-1} \frac{3^i}{4^i} \leq \frac{1}{16} + \frac{2}{\zeta_{\text{sp}}} \leq \frac{1}{10}.$$

Here we have used the fact that ζ_{sp} is a large constant.

Query Complexity. From Lemma 2 and 11, the number of `ISC` queries made by `IsCollision` (for a constant threshold $\tau = \zeta_{\text{sp}}$) and `ApproxSubset` over κ iterations is,

$$O\left(\sum_{j=1}^{\kappa} \log(2^{\kappa+5})\right) = O\left(\kappa^2\right)$$ ◀

Note that the number of iterations and the query complexity of `ApproxSequence` (as mentioned in the above lemma) are random variables. We will argue their expectation in the following lemma. Moreover, when we discuss the full algorithm in Section 5 (that uses `ApproxSequence` as a subroutine), we show that the query complexity of the full algorithm is $O(\kappa^2 \cdot \log \kappa)$, where κ is the number of iterations of `ApproxSequence`. The following lemma will be useful in establishing the final query complexity in Section 5.

▶ **Lemma 13.** $\mathbb{E}[\kappa] = O(\log m)$, $\mathbb{E}[\kappa^2] = O(\log^2 m)$, *and* $\mathbb{E}[\kappa^2 \log \kappa] = O(\log^2 m \cdot \log \log m)$.

From Lemma 12 and Lemma 13, Lemma 6 follows. In the rest of the section, we prove Lemma 13 and Lemma 6.

Proof of Lemma 13. Here, we prove that

$$\mathbb{E}\left[\kappa^2 \log \kappa\right] = O(\log^2 m \log \log m)$$

The remaining bounds follow similarly.

We want to analyse $\mathbb{E}[\kappa^2 \log \kappa]$ where the expectation is over the distribution of κ for a run of the algorithm. To analyse this, we consider two different cases. For the first case, let t^* denote the time when the algorithm generates a subgraph with less than or equal to ζ_{sp} edges. Formally, we define t^* as the time t such that $m(U_t) \leq \zeta_{\text{sp}}$, and $m(U_{t-1}) > \zeta_{\text{sp}}$. If this is not the case, i.e. when $m(U_t) > \zeta_{\text{sp}}$ holds throughout the algorithm, we show that $\kappa = O(\log m)$.

First, we bound the expectation of $m(U_t)$ for some $t \in [\kappa]$. By Lemma 7, we have

$$\mathbb{E}\left[m(U_t) \mid m(U_{t-1})\right] \leq \frac{m(U_{t-1})}{2}.$$

By a inductive argument, and the fact that $m(U_0) = m$, we have

$$\mathbb{E}[m(U_t)] = \mathbb{E}[\mathbb{E}[m(U_t) \mid m(U_{t-1})]] \leq \frac{\mathbb{E}[m(U_{t-1})]}{2} \leq \ldots \leq \frac{m}{2^t} \tag{3}$$

Now, when $m(U_j) > \zeta_{\text{sp}}$ holds throughout the algorithm, we have

$$\zeta_{\text{sp}} < \mathbb{E}\left[m(U_\kappa)\right] \leq \frac{m}{2^\kappa} \implies \kappa = O(\log m)$$

Hence, we now focus on the case when there exists a $t^* \leq \kappa$. Note that t^* is unique for a run of the algorithm. Once, the algorithm reaches t^*, the probability of the algorithm terminating depends on the calls to `IsCollision` succeeding. We start with analysing the conditional expectation of $\kappa^2 \log \kappa$ given t^*.

$$\begin{aligned}
\mathbb{E}\left[\kappa^2 \log \kappa \mid t^*\right] &= \sum_{j=1}^{\infty} j^2 \log j \Pr\left[\kappa = j \mid t^*\right] \\
&= \sum_{j=1}^{t^*-1} j^2 \log j \Pr\left[\kappa = j \mid t^*\right] + \sum_{j=t^*}^{\infty} j^2 \log j \Pr\left[\kappa = j \mid t^*\right] \\
&= \sum_{j=t^*}^{\infty} j^2 \log j \Pr\left[\kappa = j \mid t^*\right] && \text{(As } t^* \leq \kappa\text{)}
\end{aligned}$$

$$\leq \sum_{j=t^*}^{\infty} j^2 \log j \Pr[\kappa \geq j \mid t^*]$$
$$\leq \sum_{j=t^*}^{\infty} \frac{j^2 \log j}{2^{j+5}}$$
$$= O\left(t^{*2} \log t^*\right) \tag{4}$$

Now, we analyse the distribution of t^*. We use (3) to obtain a bound on the probabilities of different values of t^*.

$$\begin{aligned}\Pr[t^* = t] \leq \Pr[t^* \geq t] &\leq \Pr[m(U_t) > \zeta_{\text{sp}}] \\ &\leq \frac{\mathbb{E}[m(U_t)]}{\zeta_{\text{sp}}} && \text{Markov's Inequality (Lemma 17)} \\ &\leq \frac{m}{2^t \zeta_{\text{sp}}} && \text{Equation (3)} \end{aligned} \tag{5}$$

Now, we bound the expectation of $\kappa^2 \log \kappa$.

$$\begin{aligned}\mathbb{E}[\kappa^2 \log \kappa] &= \mathbb{E}\left[\mathbb{E}\left[\kappa^2 \log \kappa \mid t^*\right]\right] \\ &= \mathbb{E}[t^{*2} \log t^*] && \text{(Equation (4))} \\ &= \sum_{j=1}^{\infty} j^2 \log j \Pr[t^* = j] \\ &= \sum_{j=1}^{2\log m} j^2 \log j \Pr[t^* = j] + \sum_{j=2\log m+1}^{\infty} j^2 \log j \Pr[t^* = j] \\ &\leq 8 \log^2 m \log\log m \sum_{j=1}^{2\log m} \Pr[t^* = j] + \sum_{2\log m}^{\infty} \frac{m j^2 \log j}{2^j \zeta_{\text{sp}}} \\ &= O\left(\log^2 m \log\log m\right)\end{aligned}$$

which proves the desired result. ◀

Lemma 6 follows from the algorithm `ApproxSequence` (Algorithm 2) along with Lemma 12 and Lemma 13.

5 Full algorithm

In this section, we present the full algorithm for estimating the number of edges using `ISC` queries and prove our main result stated below.

▶ **Theorem 1** (Main result). *There exists a randomized algorithm that, given access to a graph G through `ISC` queries and an approximation parameter $\varepsilon \in (0, 1/3)$, outputs an estimate $\widehat{m}$ of the number of edges m in G such that, with probability at least $2/3$,*

$$(1-\varepsilon)m \;\leq\; \widehat{m} \;\leq\; (1+\varepsilon)m.$$

The algorithm uses $O(\varepsilon^{-2} \log^2 m \cdot \log\log m)$ `ISC` queries in expectation.

`EdgeEstimation` (Algorithm 4) begins by calling `ApproxSequence` which outputs an approximation preserving sequence $U_1, \ldots, U_\kappa$, where each $p_i = m(U_i)/m(U_{i-1})$ is bounded below by a constant. This ensures that each ratio can be estimated efficiently. To estimate

each p_i, we use the subroutine `MulEstimation` which returns an $(1 \pm \varepsilon)$-approximation of p_i. The procedure `MulEstimation` is described in Algorithm 3 and its guarantee is established in Lemma 14.

Algorithm 3 `MulEstimation`$(U', U, \varepsilon, \delta)$.

Require: $U' \subseteq U$ and `ISC` access to the subset $U \subseteq V$.
Ensure: $\widehat{p}(U')$
1: Uniformly and Independently sample $O\left(\log(\delta^{-1})/\varepsilon^2\right)$ random edges from U and let S be the multi-set of sampled edges.
2: **return** $\widehat{p}(U') = \frac{1}{|S|}\sum_{e\in S} \mathbf{1}_{e\in U'}$.

Algorithm 4 `EdgeEstimation`(V, ε).

Require: `ISC` access to any subset $U \subseteq V$.
Ensure: An estimate of $\widehat{m}$
1: $U_1, U_2, \cdots, U_\kappa \leftarrow$ `ApproxSequence`(V)
2: $\widehat{m}_b \leftarrow$ `BirthdayEstimate`$(U_\kappa, 8\zeta_{\text{sp}}, \frac{\varepsilon}{3})$
3: Initialize $\widehat{p}_i \leftarrow 1$, for all $i \in [\kappa]$
4: **for** $i = 1$ to κ **do**
5: $\quad \widehat{p}_i \leftarrow$ `MulEstimation`$(U_i, U_{i-1}, \frac{\varepsilon}{200\sqrt{\kappa}}, \frac{1}{16\kappa})$
6: **end for**
7: $\widehat{m} \leftarrow \widehat{m}_b / \prod_{i=1}^{\kappa} \widehat{p}_i$
8: **return** $\widehat{m}$

▶ **Lemma 14** (Correctness of `MulEstimation`). *For subsets $U' \subseteq U \subseteq V$ and $\varepsilon > 0$, If $p(U') = \frac{m(U')}{m(U)} \geq \frac{1}{32}$, then `MulEstimation` outputs $\widehat{p}(U') = (1+x)p(U')$, where $\mathbb{E}[x] = 0$ and $x \in (-\varepsilon, \varepsilon)$ with probability at least $1-\delta$. The number of `ISC` queries made is $O\left(\log(\delta^{-1})/\varepsilon^2\right)$.*

Proof. Let X_j be the indicator random variable such that,

$$X_j = \begin{cases} 1 & \text{If the } j^{th} \text{ sampled edge is in } U' \\ 0 & \text{Otherwise} \end{cases}$$

Define number of edges sampled that belong to U' is $X = \sum_{j\in[|S|]} X_j$, where S is the multi-set of sampled edges (Line 2) and the expected size of X is

$$\mathbb{E}[X] = \sum_{j\in[|S|]} \mathbb{E}[X_j] = |S| \cdot \frac{m(U')}{m(U)} = |S| \cdot p(U') \geq \frac{|S|}{32}$$

Therefore,

$$\mathbb{E}[x] = \mathbb{E}\left[\frac{\widehat{p}(U')}{p(U')}\right] - 1 = \frac{\mathbb{E}[X]}{|S|\,p(U')} - 1 = 0$$

By the Chernoff Bound (see Lemma 19), we have

$$\Pr[x \in (-\varepsilon,\varepsilon)] = \Pr\left[\widehat{p}(U') \in (1\pm\varepsilon)p(U')\right] = \Pr\left[X \in (1\pm\varepsilon)\,\mathbb{E}[X]\right] \geq 1 - 2\exp\left(-\frac{\varepsilon^2\,|S|}{96}\right) \geq 1-\delta,$$

where the last step uses $|S| = O\left(\log(\delta^{-1})/\varepsilon^2\right)$. ◀

Now, we prove the guarantees of `EdgeEstimation` In particular, we prove the correctness of `EdgeEstimation` and its query complexity in Lemma 16 and Lemma 15, respectively.

▶ **Lemma 15.** *The expected query complexity of `EdgeEstimation` is* $O(\log^2 m \cdot \log\log m)$.

Proof. From Lemma 14, the number of `ISC` queries made by `MulEstimation` in j-th iteration is

$$O\left(\frac{\log(16\kappa)}{\varepsilon'^2}\right) \text{ where } \varepsilon' = \frac{\varepsilon}{200\sqrt{\kappa}}$$

Summing over κ iterations, we get

$$O\left(\sum_{j=1}^{\kappa} \frac{\log(\kappa)}{\varepsilon'^2}\right) = O\left(\kappa^2 \frac{\log\kappa}{\varepsilon^2}\right)$$

From Lemma 12, the number of `ISC` queries made by `ApproxSequence` is $O(\kappa^2)$. Finally, from Lemma 3, the number of `ISC` queries made by `BirthdayEstimate` is $O\left(\sqrt{8\zeta_{\text{sp}}}/\varepsilon\right) = O(1/\varepsilon)$. Combining all contributions, the total number of `ISC` queries made by `EdgeEstimation` is dominated by `MulEstimation`, and hence is, $O(\kappa^2 \log\kappa/\varepsilon^2)$. From Lemma 13,

$$\mathbb{E}[\kappa^2 \log\kappa] = O(\log^2 m \log\log m)$$

which bounds the expected query complexity. ◀

▶ **Lemma 16.** *In `EdgeEstimation`, the estimate* $\widehat{m}$ *is an* $(1 \pm \varepsilon)$*-approximation to* m *with probability at least* $3/4$.

Proof. Let $\mathcal{E}_1 = \mathcal{E}_{\text{sub}} \cap \mathcal{E}_{\text{bir}}$, where $\mathcal{E}_{\text{sub}}$ be the event that `ApproxSequence` outputs an approximation-preserving sequence $U_1, U_2, \ldots, U_\kappa$ with $m(U_\kappa) \le 8\zeta_{\text{sp}}$. From Lemma 12 we have $\Pr[\mathcal{E}_{\text{sub}}] \ge 4/5$.

$\mathcal{E}_{\text{bir}}$ be the event that `BirthdayEstimate` outputs an $(1 \pm \varepsilon/3)$ approximation of $m(U_\kappa)$ from Lemma 3. Also, from Lemma 3, we have $\Pr[\mathcal{E}_{\text{bir}} \mid \mathcal{E}_{\text{sub}}] \ge 15/16$. Therefore,

$$\Pr[\mathcal{E}_1] = \Pr[\mathcal{E}_{\text{sub}}] \Pr[\mathcal{E}_{\text{bir}} \mid \mathcal{E}_{\text{sub}}] \ge 1 - \left(\frac{1}{5} + \frac{1}{16}\right) \ge \frac{59}{80}$$

Let $\mathcal{E}_2$ be the event that `MulEstimation`$(U_i, U_{i-1}, \varepsilon', 1/16\kappa)$ outputs $\widehat{p_i} = (1 \pm x_i/4)p_i$ such that $\mathbb{E}[x_i] = 0$ and $x_i \in (-4\varepsilon', 4\varepsilon')$ for all $i \in [\kappa]$, where $\varepsilon' = \frac{\varepsilon}{200\sqrt{\kappa}}$. Therefore, using Lemma 14 and applying the union bound over all $i \in [\kappa]$,

$$\Pr[\mathcal{E}_2 \mid \mathcal{E}_1] = 1 - \sum_{j=1}^{\kappa} \frac{1}{16\kappa} \ge \frac{15}{16}.$$

Let $\mathcal{E}_3$ be the event that $|\sum_{i=1}^{\kappa} x_i| \le \varepsilon/9$. By Hoeffding's bound (Lemma 20),

$$\begin{aligned}\Pr[\mathcal{E}_3 \mid \mathcal{E}_1 \cap \mathcal{E}_2] &= 1 - \Pr\left[\left|\sum_{i=1}^{\kappa} x_i\right| > \frac{\varepsilon}{9} \,\middle|\, \mathcal{E}_1 \cap \mathcal{E}_2\right] \\ &\ge 1 - 2\exp\left(-\frac{2\varepsilon^2}{81\sum_{i=1}^{\kappa}\left(\frac{2\varepsilon}{50\sqrt{\kappa}}\right)^2}\right) \ge 1 - \frac{1}{e^{12}}. \end{aligned} \tag{6}$$

Hence,

$$\Pr[\mathcal{E}_1 \cap \mathcal{E}_2 \cap \mathcal{E}_3] = \Pr[\mathcal{E}_1] \cdot \Pr[\mathcal{E}_2|\mathcal{E}_1] \cdot \Pr[\mathcal{E}_3 \mid \mathcal{E}_1 \cap \mathcal{E}_2] \geq 1 - \left(\frac{21}{80} + \frac{1}{16} + \frac{1}{e^{12}}\right) \geq \frac{2}{3}$$

In what follows, we assume that all of the events $\mathcal{E}_1$, $\mathcal{E}_2$ and $\mathcal{E}_3$ occur. Since, $\Pr[\mathcal{E}_1 \cap \mathcal{E}_2 \cap \mathcal{E}_3] \geq 2/3$, it is enough to argue that `EdgeEstimation` returns an $(1 \pm \varepsilon)$-approximation to m when all $\mathcal{E}_1$, $\mathcal{E}_2$ and $\mathcal{E}_3$ holds.

Combining the output of `MulEstimation` over the κ iterations, we get,

$$\prod_{i=1}^{\kappa} \widehat{p}_i = \prod_{i=1}^{\kappa} p_i \cdot \prod_{i=1}^{\kappa} \left(1 + \frac{x_i}{4}\right)$$

Using the fact that for each $x_i \in (0, 1)$, it can be shown that

$$1 - \frac{\sum_{i=1}^{\kappa} x_i}{4} \leq \prod_{i=1}^{\kappa} \left(1 + \frac{x_i}{4}\right) \leq 1 + \sum_{i=1}^{\kappa} x_i$$

Since event $\mathcal{E}_3$ occurs,

$$\prod_{i=1}^{\kappa} p_i \cdot \left(1 - \frac{\varepsilon}{36}\right) \leq \prod_{i=1}^{\kappa} \widehat{p}_i \leq \prod_{i=1}^{\kappa} p_i \cdot \left(1 + \frac{\varepsilon}{9}\right)$$

Using the fact that for any $\varepsilon \in (0, 1/3)$, we have $1 - 3\varepsilon \leq (1 + \varepsilon)^{-1}$ and $(1 - \varepsilon)^{-1} \leq 1 + 3\varepsilon$. We get,

$$\left(1 - \frac{\varepsilon}{3}\right) \cdot \frac{1}{\prod_{i=1}^{\kappa} p_i} \leq \frac{1}{\prod_{i=1}^{\kappa} \widehat{p}_i} \leq \frac{1}{\prod_{i=1}^{\kappa} p_i} \cdot \left(1 + \frac{\varepsilon}{12}\right) \tag{7}$$

Since $\mathcal{E}_1$ occurs, i.e., $\mathcal{E}_{\text{bir}}$ occurs, $\widehat{m}_b$ satisfies the following:

$$\left(1 - \frac{\varepsilon}{3}\right) \cdot m(U_\kappa) \leq \widehat{m}_b \leq \left(1 + \frac{\varepsilon}{3}\right) \cdot m(U_\kappa) \tag{8}$$

Combining (7) and (8), we get,

$$\left(1 - \frac{\varepsilon}{3}\right)^2 \cdot \frac{m(U_\kappa)}{\prod_{i=1}^{\kappa} p_i} \leq \widehat{m} = \frac{\widehat{m}_b}{\prod_{i=1}^{\kappa} \widehat{p}_i} \leq \left(1 + \frac{\varepsilon}{12}\right)\left(1 + \frac{\varepsilon}{3}\right) \cdot \frac{m(U_\kappa)}{\prod_{i=1}^{\kappa} p_i}$$

Since, $p_i = \frac{m(U_i)}{m(U_{i-1})}$ for each $i \in [\kappa]$ and $U_0 = V$, $\frac{m(U_\kappa)}{\prod_{i=1}^{\kappa} p_i} = m(U_0) = m$. So, simplifying the above expression, we have

$$(1 - \varepsilon)\, m \leq \widehat{m} \leq (1 + \varepsilon)\, m. \qquad \blacktriangleleft$$

References

1 Tomer Adar, Yahel Hotam, and Amit Levi. When Local and Non-Local Meet: Quadratic Improvement for Edge Estimation with Independent Set Queries, 2026. arXiv:2601.21457. doi:10.48550/arXiv.2601.21457.

2 Raghavendra Addanki, Andrew McGregor, and Cameron Musco. Non-Adaptive Edge Counting and Sampling via Bipartite Independent Set Queries. In *Proceedings of the 30th Annual European Symposium on Algorithms, ESA*, pages 2:1–2:16, 2022. doi:10.4230/LIPIcs.ESA.2022.2.

3 Matthew Aldridge, Oliver Johnson, and Jonathan Scarlett. Group Testing: An Information Theory Perspective. *Foundations and Trends in Communications and Information Theory*, 15(3-4):196–392, 2019. doi:10.1561/0100000099.

4 Maryam Aliakbarpour, Amartya Shankha Biswas, Themis Gouleakis, John Peebles, Ronitt Rubinfeld, and Anak Yodpinyanee. Sublinear-Time Algorithms for Counting Star Subgraphs via Edge Sampling. *Algorithmica*, 80(2):668–697, 2018. doi:10.1007/S00453-017-0287-3.

5 Sepehr Assadi, Deeparnab Chakrabarty, and Sanjeev Khanna. Graph Connectivity and Single Element Recovery via Linear and OR Queries. In *Proceedings of the 29th Annual European Symposium on Algorithms (ESA)*, pages 7:1–7:19, 2021. doi:10.4230/LIPIcs.ESA.2021.7.

6 Paul Beame, Sariel Har-Peled, Sivaramakrishnan Natarajan Ramamoorthy, Cyrus Rashtchian, and Makrand Sinha. Edge Estimation with Independent Set Oracles. *ACM Transactions on Algorithms*, 16(4):52:1–52:27, 2020. preliminary version of this paper appeared as an extended abstract in the proceedings of ITCS 2018. doi:10.1145/3404867.

7 Lorenzo Beretta, Deeparnab Chakrabarty, and C. Seshadhri. Faster Estimation of the Average Degree of a Graph Using Random Edges and Structural Queries. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 939–971, 2026. doi:10.1137/1.9781611978971.38.

8 Anup Bhattacharya, Arijit Bishnu, Arijit Ghosh, and Gopinath Mishra. Faster Counting and Sampling Algorithms using Colorful Decision Oracle. *ACM Transactions on Computation Theory*, 16(2):12:1–12:19, 2024. A preliminary version of this paper appeared as an extended abstract in the proceedings of STACS 2022. doi:10.1145/3657605.

9 Rishiraj Bhattacharyya and Sourav Chakraborty. Property Testing of Joint Distributions using Conditional Samples. *ACM Transactions on Computation Theory*, 10(4):16:1–16:20, 2018. doi:10.1145/3241377.

10 Rishiraj Bhattacharyya, Sourav Chakraborty, Yash Pote, Uddalok Sarkar, and Sayantan Sen. Testing Self-Reducible Samplers. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence (AAAI)*, pages 7952–7960, 2024. doi:10.1609/AAAI.V38I8.28632.

11 Arijit Bishnu, Debarshi Chanda, Buddha Dev Das, Arijit Ghosh, and Gopinath Mishra. Optimal non-adaptive algorithm for edge estimation. *CoRR*, abs/2512.11994, 2025. doi:10.48550/arXiv.2512.11994.

12 Arijit Bishnu, Debarshi Chanda, and Gopinath Mishra. Towards tight bounds for estimating degree distribution in streaming and query models. *CoRR*, abs/2507.21784, 2025. doi:10.48550/arXiv.2507.21784.

13 Arijit Bishnu, Arijit Ghosh, and Gopinath Mishra. On the Complexity of Triangle Counting Using Emptiness Queries. In *Proceedings of the 27th International Conference on Randomization and Computation (RANDOM)*, volume 275, pages 48:1–48:22, 2023. doi:10.4230/LIPIcs.APPROX/RANDOM.2023.48.

14 Clément L. Canonne, Xi Chen, Gautam Kamath, Amit Levi, and Erik Waingarten. Random Restrictions of High Dimensional Distributions and Uniformity Testing with Subcube Conditioning. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 321–336, 2021. doi:10.1137/1.9781611976465.21.

15 Clément L. Canonne, Dana Ron, and Rocco A. Servedio. Testing equivalence between distributions using conditional samples. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1174–1192, 2014. doi:10.1137/1.9781611973402.87.

16 Clément L. Canonne, Dana Ron, and Rocco A. Servedio. Testing Probability Distributions using Conditional Samples. *SIAM Journal on Computing*, 44(3):540–616, 2015. An extended abstract of this work appeared in SODA 2014. doi:10.1137/130945508.

17 Diptarka Chakraborty, Sourav Chakraborty, Gunjan Kumar, and Kuldeep S. Meel. Approximate Model Counting: Is SAT Oracle More Powerful Than NP Oracle? In *Proceedings of the 50th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 261, pages 123:1–123:17, 2023. doi:10.4230/LIPIcs.ICALP.2023.123.

18 Sourav Chakraborty, Eldar Fischer, Yonatan Goldhirsh, and Arie Matsliah. On the power of conditional samples in distribution testing. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science (ITCS)*, pages 561–580, 2013. doi:10.1145/2422436.2422497.

19 Debarshi Chanda. Improved Bounds with a Simple Algorithm for Edge Estimation for Graphs of Unknown Size. *CoRR*, abs/2511.03650, 2025. doi:10.48550/arXiv.2511.03650.

20 Debarshi Chanda, Buddha Dev Das, Arijit Ghosh, and Gopinath Mishra. From decision to random certificates: Exponential separation for edge estimation with independent set queries, 2026. arXiv:2607.07483.

21 Xi Chen, Amit Levi, and Erik Waingarten. Nearly Optimal Edge Estimation with Independent Set Queries. In *Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2916–2935, 2020. doi:10.1137/1.9781611975994.177.

22 Fan Chung and Linyuan Lu. Concentration Inequalities and Martingale Inequalities: A Survey. *Internet Mathematics*, 3(1):79–127, 2006.

23 Holger Dell, John Lapinskas, and Kitty Meeks. Approximately Counting and Sampling Small Witnesses Using a Colorful Decision Oracle. *SIAM Journal on Computing*, 51(4):849–899, 2022. A preliminary version of this paper appeared as an extended abstract in the proceedings of SODA 2020. doi:10.1137/19M130604X.

24 Holger Dell, John Lapinskas, and Kitty Meeks. Nearly Optimal Independence Oracle Algorithms for Edge Estimation in Hypergraphs. In *Proceedings of the 51st International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 54:1–54:17, 2024. doi:10.4230/LIPIcs.ICALP.2024.54.

25 Robert Dorfman. The Detection of Defective Members of Large Populations. *The Annals of Mathematical Statistics*, 14(4):436–440, 1943.

26 Talya Eden, Dana Ron, and C. Seshadhri. Sublinear Time Estimation of Degree Distribution Moments: The Arboricity Connection. *SIAM Journal on Discrete Mathematics*, 33(4):2267–2285, 2019. doi:10.1137/17M1159014.

27 Talya Eden and Will Rosenbaum. Lower Bounds for Approximating Graph Parameters via Communication Complexity. In *Proceedings of the 21st International Conference on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 11:1–11:18, 2018. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.11.

28 Uriel Feige. On Sums of Independent Random Variables with Unbounded Variance and Estimating the Average Degree in a Graph. *SIAM Journal on Computing*, 35(4):964–984, 2006. An extended abstract of this paper appeared in the proceedings of STOC 2004. doi:10.1137/S0097539704447304.

29 Oded Goldreich and Dana Ron. Approximating Average Parameters of Graphs. *Random Structures & Algorithms*, 32(4):473–493, 2008. An extended abstract of this paper appeared in the proceedings of RANDOM 2006. doi:10.1002/RSA.20203.

30 Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, Cambridge, 2005. doi:10.1017/CBO9780511813603.

31 Dana Ron and Gilad Tsur. The Power of an Example: Hidden Set Size Approximation Using Group Queries and Conditional Sampling. *ACM Transactions on Computation Theory*, 8(4), 2016. doi:10.1145/2930657.

32 Larry J. Stockmeyer. The Complexity of Approximate Counting (Preliminary Version). In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing (STOC)*, pages 118–126, 1983. doi:10.1145/800061.808740.

33 Larry J. Stockmeyer. On Approximation Algorithms for #P. *SIAM Journal on Computing*, 14(4):849–861, 1985. doi:10.1137/0214060.

34 Jakub Tětek and Mikkel Thorup. Edge Sampling and Graph Parameter Estimation via Vertex Neighborhood Accesses. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1116–1129, 2022.

A Concentration Inequalities

We use the following concentration bounds.

▶ **Lemma 17** (Markkov's inequlaity, see [30]). *Let X be a random variable that assumes only nonnegative values. Then, for all $a > 0$*

$$\Pr[X \geq a] \leq \frac{\mathbb{E}[X]}{a}.$$

▶ **Lemma 18** (Chebyshev's inequality, see [30]). *Let X be a random variable and $a > 0$, Then*

$$\Pr[|X - \mathbb{E}[X]| > a] \leq \frac{\mathrm{Var}(X)}{a^2}.$$

▶ **Lemma 19** (Multiplicative Chernoff bound, see [30]). *Let $X_1, X_2, ..., X_t$ be i.i.d. random variables where $\Pr[X_i = 1] = p$ and $\Pr[X_i = 0] = 1 - p$ for all $i \in [t]$, and $X = \sum_{i\in[t]} X_i$. Then, we have*

$$\Pr[X \leq (1-\varepsilon)\,\mathbb{E}[X]] \leq \exp\left(-\frac{\varepsilon^2\,\mathbb{E}[X]}{3}\right) \qquad 0 \leq \varepsilon < 1$$

$$\Pr[X \geq (1+\varepsilon)\,\mathbb{E}[X]] \leq \exp\left(-\frac{\varepsilon^2\,\mathbb{E}[X]}{2}\right) \qquad \varepsilon \geq 0$$

▶ **Lemma 20** (Hoeffding's bound, see [30]). *Let $X_1, X_2, \ldots, X_t$ be independent random variables such that for all $i \in [t]$, $\mathbb{E}[X] = \mu$ and $\Pr[a \leq X_i \leq b] = 1$. Then*

$$\Pr\left[\left|\frac{1}{t}\sum_{i=1}^{t} X_i - \mu\right| \geq \varepsilon\right] \leq 2\exp\left(-\frac{2t\varepsilon^2}{(b-a)^2}\right).$$

▶ **Lemma 21** (See [22]). *Let f be any function of t independent random variables $Y_1, \ldots, Y_t$ and let $X_i = \mathbb{E}[f(Y_1, \ldots, Y_t)|Y_1, \ldots, Y_i]$ for $i \in [t]$ and $X_0 = \mathbb{E}[f(Y_1, \ldots, Y_t)]$. Let $\mathcal{B}$ be the event that there exists $i \in [t]$ such that $|X_i - X_{i-1}| > c_i$ where $c_1, \ldots, c_t$ are some non negative numbers. Let $S = \sum_{i=1}^{t} c_i^2$, then*

$$\Pr[|X_t - X_0| > s] \leq 2\exp\left(-\frac{s^2}{2S}\right) + \Pr[\mathcal{B}].$$

Glauber Dynamics for Random Field Ising Models on Bounded Degree Graphs

Yi Han
Institute for Advanced Study, Princeton, NJ, USA

Abstract

We study the ferromagnetic random field Ising model (RFIM) on a graph $G = (V, E)$ having maximal degree Δ, where the external field at each vertex is an i.i.d. random variable. When the random field distribution is sufficiently anti-concentrated, we prove that with high probability over the quenched randomness of the external field, the Glauber dynamics of this RFIM mixes in polynomial time as a consequence of a Poincaré inequality. This model is relevant to the Griffiths phase where the correlations decay exponentially fast in expectation over the quenched random field, but contraction does not hold point-wise due to the existence of weak fields that lead to low-temperature behavior. Previously, fast mixing of Glauber dynamics under large disorder was only proven on the integer lattice, and for RFIM on general graphs, only a sampling algorithm based on self-avoiding walks was known.

2012 ACM Subject Classification Mathematics of computing → Markov-chain Monte Carlo methods; Theory of computation → Random walks and Markov chains

Keywords and phrases Random field Ising model, Glauber dynamics, Poincaré inequality, Markov chain mixing, bounded degree graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.42

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/pdf/2605.02002`

Funding *Yi Han*: Yi Han receives a fellowship from IAS provided by the S.S. Chern Foundation for Mathematical Research Fund and the Fund for Mathematics.

1 Introduction

Consider a finite graph $G = (V, E)$, the random field Ising model (RFIM) on G at inverse temperature $\beta > 0$ is defined as the following random probability measure on $\{-1, +1\}^V$:

$$\mu_G(\sigma) \propto \exp\{H(\sigma)\}, \quad \sigma \in \{-1, +1\}^V, \tag{1.1}$$

where H is the following Hamiltonian

$$H(\sigma) = \beta \sum_{(u,v)\in E} \sigma_u \sigma_v + \sum_{u\in V} h_u \sigma_u. \tag{1.2}$$

Here $h = (h_u)_{u\in V}$ are i.i.d. random variables. The vector h is called the external field, serving as a quenched source of disorder in this model.

There has been much recent attention on RFIM on (subgraphs of) the integer lattice $\mathbb{Z}^d$ in the statistical physics literature. In $d = 2$, the existence of an arbitrarily small random field is predicted by Imry and Ma [20] to suppress the low temperature phase of the zero field ($h = 0$) Ising model, so that uniqueness and decay of correlations hold for any $\beta > 0$. The prediction was confirmed by Aizenman and Wehr [1], who proved that the expected influence on the spin at the origin by different boundary conditions at distance r away should decay with r. For the Gaussian case $h_u \sim \mathcal{N}(0, \sigma^2)$, the breakthrough work [12] confirmed

© Yi Han;
licensed under Creative Commons License CC-BY 4.0

that the influence decays exponentially in r at all $\beta > 0$ and $\sigma > 0$. This no longer holds for RFIM with dimension $d \geq 3$ and small disorder, where there is a phase transition as β varies. For small disorder, there is no decay of influence when $\beta = \infty$ [19] and when β sufficiently large [3], and this is recently proven throughout the entire low temperature regime [11].

Beyond the integer lattice, the RFIM on a general graph G has also attracted interest from a computational complexity perspective. For a large $\beta > 0$ and general external field h, estimating the partition function of μ_G up to a multiplicative error is a #BIS-hard problem in the worst case [4], and polynomial-time approximation algorithms are not expected to exist. However, the random field may turn this hard computational problem into a feasible one: for bounded degree graphs G and large enough σ where $h_u \sim \mathcal{N}(0, \sigma^2)$, [18] constructed a polynomial time approximation algorithm for the partition function of μ_G over a $1 - o(1)$ probability realization of the external fields h.

For a general graph G, it becomes much less tractable to study the influence decay of a RFIM with small or moderate disorder h_u (that is, smaller σ if $h_u \sim \mathcal{N}(0, \sigma^2)$). This is because current RFIM papers on $\mathbb{Z}^d$ that capture the influence decay at $d = 2$ or the phase change at $d = 3$ use the geometry of $\mathbb{Z}^d$ in a fundamental way and such a delicate description is hard to expect on a general graph G. As a natural next step, in this paper we study effective sampling algorithms for the RFIM μ_G on a general graph G. For sufficiently large disorder (say, larger σ), it is straightforward to verify the decay of correlations for this RFIM via a percolation argument.

In contrast, even for relatively large disorder, the RFIM still possesses major difficulties from a sampling perspective. We consider the Glauber dynamics for RFIM, where we recall that Glauber dynamics (introduced in [16]; see also [22] for more background) is a discrete time random process $(X_n(v))_{v \in V, n \in \mathbb{N}_+}$ reversible w.r.t. μ_G, that at each step n selects a vertex $v \in V$ uniformly with probability $\frac{1}{|V|}$ and then resamples $X_n(v)$ according to the conditional distribution

$$X_n(v) \sim \mu_G(\sigma_v \in \cdot \mid (\sigma_w)_{w \neq v} = (X_{n-1}(w))_{w \neq v}), \tag{1.3}$$

and where we set $X_n(w) = X_{n-1}(w)$ for all $w \neq v$. If we assume that all the $|h_u|$ are uniformly larger than some fixed constant, then it is straightforward to show that the Glauber dynamics for RFIM mixes in time $O(n \log n)$, as in [18], Theorem 7. However, in the typical RFIM setting, although most external fields h_u are large, there are large regions where the absolute values of external fields $|h_u|$ are small, and the Glauber dynamics does not contract at these vertices (see [18], Remark 7). To avoid this problem, [18] constructed a polynomial sampling algorithm for RFIM μ_G on general bounded degree graphs G utilizing the Weitz self-avoiding walk tree [24], which is a global algorithm that differs in spirit from the Glauber update rule. As a future direction, [18] asked whether one can use Markov chain based algorithms (such as Glauber dynamics) to effectively sample from RFIM.

For Glauber dynamics of RFIM on $\mathbb{Z}^d, d \geq 2$, [14] conducted a detailed study for the mixing time of μ_G and designed a sampling algorithm at weak disorder. Under a notion of weak spatial mixing that measures averaged influence decay within a ball, they constructed a polynomial time sampling algorithm for RFIM based on a weak Poincaré inequality for Glauber dynamics. Under a stronger notion of influence decay and in particular when random fields have large disorder (h_u have good anti-concentration), they prove that Glauber dynamics for μ_G mixes in polynomial time as a corollary of a full Poincaré inequality. However, the proof in [14] is highly dependent on the lattice structure of $\mathbb{Z}^d$ due to a coarse graining argument in Section 5, and in Section 4 also somewhat relies on a volume growth exponent on G that rules out graphs of exponential volume growth. Therefore, the Glauber dynamics of μ_G on general graphs G had remained open.

The main result of this paper is a Poincaré inequality for μ_G which implies polynomial time mixing for μ_G.

Let $\mathcal{E}_G$ denote the Dirichlet form of the (discrete time) Glauber dynamics, so that for any test function $\varphi : \{-1,+1\}^V \to \mathbb{R}$,

$$\mathcal{E}_G(\varphi,\varphi) := \frac{1}{2|V|}\sum_{\sigma\sim\sigma'}\frac{\mu_G(\sigma)\mu_G(\sigma')}{\mu_G(\sigma)+\mu_G(\sigma')}(\varphi(\sigma)-\varphi(\sigma'))^2, \tag{1.4}$$

so that the summation is taken over pairs $\sigma,\sigma' \in \{-1,1\}^V$ differing only at one coordinate. Then we define the spectral gap of the chain via

$$\mathbf{gap}_G := \inf_{\varphi}\frac{\mathcal{E}_G(\varphi,\varphi)}{\mathrm{Var}_G(\varphi)} \tag{1.5}$$

with the infimum taken over all test functions $\varphi : \{-1,1\}^V \to \mathbb{R}$ with non-zero variance.

1.1 Polynomial time mixing for Glauber dynamics of RFIM with large disorder

Before stating the main technical result, we outline the anti-concentration assumptions on the external field h in the following:

▶ **Assumption 1.1.** *Let $\Delta \geq 3$ and $p_0 \in (0,1)$ be a constant such that $p_0(\Delta-1) < 1$, let $K = K(p_0) > 0$ be a constant such that*

$$\rho = \rho(\beta,\Delta,K) := \frac{e^{\Delta\beta-K}}{e^{\Delta\beta-K}+e^{-\Delta\beta+K}}, \tag{1.6}$$

is less than $\frac{p_0}{4}$. Suppose that the RFIM μ_G with external field h is such that $(h_u)_{u\in V}$ are i.i.d. and such that

$$\mathbb{P}_h(|h_u| \leq K) < \frac{p_0}{2}.$$

We then denote by

$$\alpha_* = \frac{1}{2}\log\frac{(\Delta-2)^{\Delta-2}}{p_0(\Delta-1)^{\Delta-1}(1-p_0)^{\Delta-2}}.$$

Note that for fixed $\Delta > 0$, $\alpha_* \uparrow \infty$ as $p_0 \downarrow 0$.

For a graph $G = (V,E)$, we use $|V|$ for the number of vertices in G.

Our first main result on Glauber dynamics for RFIM for any bounded degree graph G is as follows:

▶ **Theorem 1.2.** *Let $G = (V,E)$ be a graph of degree at most Δ. Assume that Assumption 1.1 is satisfied for some $p_0 \in (0,1)$, then with $\mathbb{P}_h$- probability at least $1 - O_{\beta,\Delta,p_0}(\frac{1}{|V|})$ over the external field (here O_{β,Δ,p_0} denotes constants that may depend on β, Δ and p_0), the spectral gap of RFIM measure μ_G satisfies*

$$\mathbf{gap}_G \geq |V|^{-1}\exp\left(-\frac{16\beta\Delta\ln|V|}{\alpha_*}\right),$$

and the ϵ- mixing time of Glauber dynamics for μ_G is

$$t_{mix}(P,\epsilon) = O\left(|V|^{1+\frac{16\beta\Delta}{\alpha_*}}\left(\beta\Delta|V| + \|h\|_1 + \log(1/\epsilon)\right)\right).$$

Here $\|h\|_1 := \sum_{x\in V}|h_x|$.

▶ Remark 1.3. One may also consider the following more general model with different edge weights β_{uv}. Suppose that for each edge $(u,v) \in E$ we assign a constant $\beta_{uv} \in \mathbb{R}$ such that $|\beta_{uv}| \leq \beta$. Consider the following Hamiltonian $\widetilde{H}$ and Ising model $\mu_{\widetilde{G}}$:

$$\widetilde{H}(\sigma) = \sum_{(u,v)\in E} \beta_{uv}\sigma_u\sigma_v + \sum_{u\in V} h_u\sigma_u, \quad \mu_{\widetilde{G}}(\sigma) \approx \exp\{\widetilde{H}(\sigma)\}, \quad \sigma \in \{-1,1\}^V. \tag{1.7}$$

In the ferromagnetic case $\beta_{uv} \geq 0$ with nonconstant edge weights, the same strategy should be adaptable by using edge-dependent tilting rates. We do not pursue this extension here. However, when some $\beta_{uv} < 0$, the edge-tilt localization scheme that we will use breaks down, so Theorem 1.2 does not readily extend to non-monotone systems.

▶ Remark 1.4. We note that on the integer lattice $\mathbb{Z}^d$, [14], Section 5 obtained a mixing time $O(|V|^{1+o(1)})$ for a high probability realization of h via a coarse graining argument (they considered a continuous time Glauber dynamics where vertices are updated at rate 1, and we rewrite their runtime in the discrete setting). We consider much more general graphs (such as expander graphs), and by setting $p_0 > 0$ sufficiently small (so that $\frac{16\Delta\beta}{\alpha_*} \leq \epsilon$) we can achieve $O(|V|^{2+\epsilon})$ mixing time for any $\epsilon > 0$. Note that the spectral gap scales like $|V|^{-1-\epsilon}$ when $p_0 > 0$ is small, which is almost optimal up to the $\epsilon > 0$.

We also note that the anti-concentration condition is designed for a worst case percolation argument to work through, and thus may not be the weakest possible condition. Previous papers concerning a large external field also work under these types of conditions, see [18] and [14], Section 5.

1.2 Overview of the proof

The RFIM measure μ_G is difficult to study by itself since it does not satisfy the Dobrushin condition or standard notions of spectral independence when β is large. Motivated by the recent application of localization schemes of Chen and Eldan [9], we will seek a continuous (and stochastic) deformation of μ_G to a target measure that is already known to satisfy a functional inequality such as Poincaré inequality, and that the deformation approximately preserves certain statistics, such as variance, of μ_G.

How can we choose a proper localization scheme for RFIM? The first candidate choice is the stochastic localization scheme in [9] where we use a Brownian motion on the localizing path. The exact path of localization still requires a delicate choice: one standard deformation is to use the localization path to weaken the quadratic interaction, so that μ_G is deformed to

$$\nu_t^1(\sigma) \propto \exp\{\beta(1-t)\sum_{(u,v)\in E} \sigma_u\sigma_v + \sum_{u\in V}(h_u + y_u^1(t))\sigma_u\}, \quad \forall\sigma \in \{-1,+1\}^V,$$

where $(y_u^1(t))_{u\in V}$ is a random external field defined by stochastic integrals of the adjacency matrix of G with respect to a Brownian motion, see [9], Fact 14 for the formula. This deformation, while useful for many important models (cf. [15], [9]), does not work well for RFIM μ_G because the external field h_u is changed to a random external field $h_u + y_u^1(t)$ that differs at each time t, and the quadratic interaction is also changed along the path. We find it hard to uniformly control ν_t^1 over all realizations of the stochastic localization path $y_u^1(t)$ with respect to **a single quenched realization** of the external field $(h_u)_{u\in V}$. In contrast, in the RFIM setting the authors of [14] chose another stochastic localization path that fixes the quadratic interaction and changes only the external field. Then μ_G deforms to

$$\nu_t^2(\sigma) \propto \exp\{\beta\sum_{(u,v)\in E} \sigma_u\sigma_v + \sum_{u\in V}(h_u + y_u^2(t))\sigma_u\}, \quad \forall\sigma \in \{-1,+1\}^V.$$

The distribution of this random external field $y_u^2(t)$ follows $t\sigma^* + B_t$ where B_t is a standard Brownian motion at time t and σ^* is independently sampled from $\sigma^* \sim \nu_0^2$. In this second localization, although the external field gains more randomness, the quadratic form is unchanged and the property WSM(C) can be passed along the localization path in a high probability sense (but not for every realization of the localization path). Thanks to this high probability preservation of WSM(C), the authors of [14] can transfer functional inequalities for ν_T^2 for a large T back to weak functional inequalities for μ_G. This idea is called virtually increasing the external field since the measure ν_T^2 is a RFIM with the same quadratic term but having a much stronger disorder in the random field. Via this strong disorder, they used a coarse graining idea on $\mathbb{Z}^d$ to verify that, for a sufficiently large T, the measure ν_T^2 satisfies a Poincaré inequality. This transfers back into weak Poincaré inequalities for μ_G on $\mathbb{Z}^d$, which shows rapid mixing of Glauber dynamics from a warm start. Then they constructed a polynomial time sampling algorithm from μ_G via Glauber dynamics on increasingly large subsets of the graph. However, the final step of coarse graining of ν_T^2 significantly restricts the geometry of G where the algorithm converges.

We take a different view towards designing localizing schemes of RFIM measures on general graphs. We need a localization path that (1) maintains a good control of the relevant covariance/correlation matrix of ν_t along the localization path, and the estimate holds for any possible pinning with respect to a single quenched external field h; and (2) the terminal measure is easy to analyze. The very recent paper [8] introduced an edge field localization scheme which is a piecewise deterministic jump Markov process that does not use SDEs. Notably, this localization path does not change the external field of RFIM and allows a quenched probability control over h. The construction is as follows. We first re-parametrize RFIM as a probability measure on $\{0,1\}^V$ rather than $\{-1,+1\}^V$. Then for any target measure μ on $\{0,1\}^V$, proceed as follows. Pick a family of events on the configuration $\sigma \in \{0,1\}^V$: $A_{(u,v)} := \{\sigma_u = \sigma_v = 1\}$ for each edge $(u,v) \in E$. Let $\mathcal{A}$ denote the collection of these events $A_{(u,v)}$. For the configuration σ denote by

$$Z_{A_{(u,v)}}(\sigma) = \mathbf{1}_{\sigma \in A_{(u,v)}}, \quad Z(\sigma) = \{A_{(u,v)} \in \mathcal{A} : \sigma \in A_{(u,v)}\}.$$

We first sample $X \sim \mu$, then for each $A_{(u,v)} \in \mathcal{A}$ we independently sample a uniform random variable $U_{(u,v)} \sim \mathrm{Unif}[0,1]$. Then we define the set of events revealed at time t:

$$T_t := \{A_{(u,v)} \in \mathcal{A} : X \in A, U_{A_{(u,v)}} \le t\}.$$

Then the localization path is the following posterior distribution, for each $S \subset \mathcal{A}$:

$$\mu_t^S(\sigma) = \mathbb{P}(X = \sigma \mid T_t = S) \propto \mu(\sigma) \cdot (1-t)^{|Z(\sigma)|} \cdot \mathbf{1}[S \subset Z(\sigma)].$$

Then observe that μ_t^S is still an Ising model on G with vertices along edges that are associated to an event in S pinned at 1, and the quadratic interactions reduce from 4β to $4\beta + \ln(1-t)$. It remains to upper bound the operator norm of a certain correlation matrix of μ_t^S uniformly over all pinnings on S and uniformly over all quadratic interaction coefficients $4\beta + \ln(1-t)$ whenever it is within $[0, 4\beta]$. As the random external fields h are unchanged along the path, we can apply a percolation argument and take a grand coupling of all these tilted RFIM models to deduce, with very high probability over h, a uniform upper bound for operator norm of the correlation matrix. Finally, we take the terminal time $\theta_* = 1 - \exp(-4\beta)$, where the conditional measure is a product measure for any pinning S and approximate tensorization of variance directly follows. This approach only relies on a percolation argument and completely bypasses fine local geometries of $\mathbb{Z}^d$. Unlike other terminal-time choices

based on high-temperature tensorization, the current edge-field denoising flow is stopped when the posterior becomes a product measure; therefore, no separate high-temperature case results are needed.

1.3 Facts, notations and conventions

1.3.1 Graph distance and boundary

Let $G = (V, E)$ be a graph. For any two vertices $u, v \in V$ we use $d(u, v)$ to denote the graph distance between u and v in G.

For a subset $A \subset V$, denote by $\partial A := \{v \in G : v \notin A, \exists w \in A, (v, w) \in E\}$ the boundary of A. For any fixed configuration $\tau \in \{-1, 1\}^{\partial A}$ on ∂A, we denote by μ_A^τ the restriction of the Ising model μ_G to A with the boundary condition τ.

Let $(u, v) \in E$ be an edge of G, for any $l \in \mathbb{N}_+$ we denote by $B((u, v), l)$ the l-neighborhood of the edge (u, v):

$$B((u, v), l) := \{w \in G : d(w, u) \le l \text{ or } d(w, v) \le l\} = B(u, l) \cup B(v, l). \tag{1.8}$$

1.3.2 Markov chain mixing times

Let P be a Markov operator on Ω with invariant measure ν, P being reversible with respect to ν. Let μ be an initial distribution that is absolutely continuous with respect to ν.

The total-variation mixing time is defined by

$$t_{mix}(P, \epsilon; \mu) = \min\{t > 0; |P^t[\mu](A) - \nu(A)| \le \epsilon, \forall A \subset \Omega\},$$

and we set

$$t_{mix}(P, \epsilon) = \max_{x \in \Omega} t_{mix}(P, \epsilon, \delta_x).$$

Then we have the following standard fact, see, for example, [21, Chapters 12–13]:

▶ **Fact 1.5.** *Assume that for all $x \in \Omega$, $\nu(\{x\}) \ge \eta$. Then*

$$t_{mix}(P, \epsilon) \le C\mathbf{gap}(P)^{-1}(\log(1/\eta) + \log(1/\epsilon)),$$

and

$$t_{mix}(P, \epsilon) \le C\rho_{LS}(P)^{-1}(\log\log(1/\eta) + \log(1/\epsilon)).$$

1.3.3 FKG and coupling

If we assume that all the RFIM interactions are positive, then μ_G satisfies useful correlation inequalities including the FKG inequality. That is, for a subset $A \subset V$ and a non-decreasing function $\varphi : \{-1, 1\}^A \to \mathbb{R}$, the mean of φ is monotone in the boundary condition: $\mu_A^\tau(\varphi) \le \mu_A^{\tau'}(\varphi)$ whenever boundary conditions satisfy $\tau \le \tau'$ pointwise.

Again assuming monotonicity, for the two boundary conditions $\tau \le \tau'$, it is standard to use the monotone coupling to couple μ_A^τ and $\mu_A^{\tau'}$. Specifically, we fix an ordering $v_1, \cdots, v_k$ of vertices in A and we then draw uniform random variables $U_1, \cdots, U_k$ independently. Then the coupling $(\sigma^\tau, \sigma^{\tau'})$ can be constructed sequentially where for each $i \ge 1$ we shall sample $\sigma_{v_i}^\tau$, $\sigma_{v_i}^{\tau'}$ from the conditional distributions given $(\sigma_{v_j}^\tau)_{j<i}$, τ and $(\sigma_{v_j}^{\tau'})_{j<i}, \tau'$ where we use the common randomness in U_i. The coupling is clearly monotone in the boundary conditions: $\sigma^\tau \le \sigma^{\tau'}$ whenever $\tau \le \tau'$.

Without assuming monotonicity, let $A \subset V$ and consider a class of Ising measures $\mu_1, \mu_2, \cdots, \mu_t$ on A with boundary conditions $\tau_1, \cdots, \tau_t$ on ∂A, where each Ising measure may have different interactions and external fields, and each μ_i may be further pinned at some $A_i \subset A$ by some value $\tau_i' \in \{-1,1\}^{A_i}$. Then we can use the same uniform random variable to couple $\mu_1, \cdots, \mu_t$ as above. Specifically, we again fix an ordering $v_1, \cdots, v_k$ of A and draw uniform random variables $U_1, \cdots, U_k$ independently. Then we construct a coupling of $(\sigma^1, \cdots, \sigma^t)$ where $\sigma^1 \sim \mu_1, \cdots, \sigma^t \sim \mu_t$ sequentially so that for each $i \geq 1$ we sample $\sigma^1_{v_i}, \cdots, \sigma^t_{v_i}$ from their respective conditional distributions given $(\sigma^r_{v_j})_{j<i}, 1 \leq r \leq t$ and $\tau_1, \cdots, \tau_t$, using U_i as common randomness. If v_i is already pinned in μ_r then we output the pinned value for $\sigma^r_{v_i}$.

1.4 Change of coordinates for RFIM

The current RFIM measure is defined on $\{-1,+1\}^V$ but we will frequently change its coordinates to a measure on $\{0,1\}^V$. The following fact summarizes the change of coordinates:

▶ **Fact 1.6.** *Let μ_G be the RFIM measure* (1.1) *defined on $\{-1,+1\}^V$. Then it admits an equivalent expression*

$$\mu_G(\sigma') \propto \exp\{4\beta \sum_{(u,v)\in E} \sigma'_u\sigma'_v + \sum_u (2h_u - 2\beta d_u)\sigma'_u\}, \quad \forall \sigma' \in \{0,1\}^V,$$

where $\sigma'_u = \frac{\sigma_u+1}{2}$ for each $u \in V$. Here d_u is the degree of u in G for each $u \in V$. Similarly, the RFIM measure $\mu_{\widetilde{G}}$ defined in (1.7) *admits an equivalent expression*

$$\mu_{\widetilde{G}}(\sigma') \propto \exp\{4 \sum_{(u,v)\in E} \beta_{uv}\sigma'_u\sigma'_v + \sum_u (2h_u - 2 \sum_{v:(u,v)\in E} \beta_{uv})\sigma'_u\}, \quad \forall \sigma' \in \{0,1\}^V.$$

Namely, after the change of coordinates, the external fields are shifted by a deterministic, vertex dependent constant but remain independent.

2 Edge-field localization schemes and spectral independence

In this section we introduce the general framework of edge tilted localization process of [8]. Then we introduce an edge field localization process for the RFIM μ_G and provide criteria for its spectral stability.

2.1 Entropy conservation and localization schemes

Consider a domain $D \subseteq \mathbb{R}$ and a convex function $\phi : D \to \mathbb{R}$. Consider μ a distribution on a finite set Ω and sample $X \sim \mu$. For any real-valued function $f : \Omega \to \mathbb{R}$, define the ϕ-entropy of f with respect to μ by

$$\mathrm{Ent}^\phi_\mu[f] := \mathrm{Ent}^\phi[f(X)] := \mathbb{E}[\phi(f(X))] - \phi(\mathbb{E}[f(X)]).$$

If ν is a probability measure on Ω absolutely continuous with respect to μ, we define the ϕ-divergence between ν and μ via

$$D_\phi(\nu \,||\, \mu) := \mathrm{Ent}^\phi_\mu\left[\frac{\nu}{\mu}\right].$$

In this paper we focus on the following two specific cases

- $\phi(x) = \chi^2(x) := x^2$, so that χ^2-entropy of f is equivalent to the variance of f, and
- $\phi(x) = \mathbf{KL}(x) := x \log x$, which induces the KL-divergence

$$D_{KL}(\nu \,||\, \mu) = \sum_{\sigma\in\Omega} \nu(\sigma) \log \frac{\nu(\sigma)}{\mu(\sigma)}.$$

We now introduce some notations from [8] on the conservation and decay of entropy for a Markov operator. To be consistent with the notations in [8], we use the state space $\{0,1\}^n$ rather than $\{-1,1\}^n$. The formula for the RFIM measure μ_G on the new coordinate $\{0,1\}^n$ is explicitly written in Fact 1.6.

▶ **Definition 2.1** (Approximate tensorization of ϕ-entropy). *We say that a distribution μ on $\{0,1\}^n$ satisfies K-approximate tensorization of ϕ-entropy if for any function $f : \Omega \to \mathbb{R}$ and any $X \sim \mu$,*

$$\mathrm{Ent}^{\phi}[f(X)] \leq K \cdot \sum_{i=1}^{n} \mathbb{E}[\mathrm{Ent}^{\phi}[f(X) \mid X_{-i}]],$$

where we denote by X_{-i} the configuration X restricted to $[n] \setminus \{i\}$.

▶ **Definition 2.2** (Decay of entropy). *Consider a Markov chain $(X_t)_{t\geq 0}$ on Ω having transition matrix P and stationary distribution μ. Then we say that it satisfies decay of ϕ-entropy with rate κ if for all functions $f : \Omega \to \mathbb{R}$,*

$$\mathrm{Ent}_{\mu}^{\phi}[Pf] \leq (1-\kappa)\,\mathrm{Ent}_{\mu}^{\phi}[f].$$

Then we introduce the general notion of a localization process in [8]. We will later specify the exact localization scheme that we shall use.

▶ **Definition 2.3.** *Consider μ a probability distribution on Ω. Then a localization scheme having μ as target distribution is made up of a pair of continuous time processes:*

- *The noising process $(X_t)_{t\in[0,1]}$. This is a Markov process such that $X_0 \sim \mu$ and X_1 follows a Dirac measure.*
- *A denoising process $(Y_t)_{t\in[0,1]}$. This process is defined as the reversal of $(X_t)_{t\in[0,1]}$: $Y_\theta = X_{1-\theta}$. Thus Y_0 is a Dirac measure and $Y_1 \sim \mu$.*

Since X_t is a time reversal of Y_t, we use the denoising process $(Y_t)_{t\in[0,1]}$ to represent the localization scheme and call it the localization process.

We would like the localization path Y_t to have the following nice property:

▶ **Definition 2.4** (Approximate conservation of ϕ-entropy). *For a given denoising process $(Y_t)_{t\in[0,1]}$ and $\theta \in [0,1]$, the process $(Y_t)_{t\in[0,1]}$ satisfies R-conservation of ϕ-entropy up to time θ if for all functions $f : \Omega \to \mathbb{R}_+$,*

$$\mathrm{Ent}^{\phi}[f(Y_1)] \leq R \cdot \mathbb{E}[\mathrm{Ent}^{\phi}[f(Y_1) \mid Y_\theta]].$$

This general notion of localization schemes gives rise to a boosting of mixing properties from a parameter regime where mixing is known, to a new parameter regime where we expect to prove mixing.

▶ **Lemma 2.5** ([6], Lemma 3.13). *Consider $(Y_t)_{t\in[0,1]}$ a denoising process for a distribution μ. Assume we can find some $\theta \in [0,1]$ such that*

1. *The conditional distribution $\mathrm{Law}(Y_1 \mid Y_\theta)$ satisfies K-approximate tensorization of ϕ-entropy.*
2. *The denoising process $(Y_t)_{t\in[0,1]}$ satisfies R-approximate conservation of ϕ-entropy up to time θ.*

Then μ satisfies (KR)-approximate tensorization of ϕ-entropy.

The following property is very useful in verifying approximation conservation of ϕ-entropy:

▶ **Definition 2.6.** *Consider the denoising process $(Y_t)_{t\in[0,1]}$ for a measure μ on Ω. Then the process $(Y_t)_{t\in[0,1]}$ is said to be ϕ-entropically stable with rate C at $\theta \in [0,1]$ if for any given function $f : \Omega \to \mathbb{R}$, denoting by $F = f(Y_1)$, then for any configuration T with $\mathbb{P}(Y_\theta = T) > 0$, we have*

$$\lim_{h\to 0^+} \frac{1}{h} \operatorname{Ent}^\phi[\mathbb{E}[F \mid Y_{\theta+h}] \mid Y_\theta = T] \le \frac{C}{1-\theta} \operatorname{Ent}^\phi[F \mid Y_\theta = T].$$

The property of ϕ-entropic stability implies approximate conservation of ϕ-entropy, as outlined in the following lemma:

▶ **Lemma 2.7** ([6], Theorem 3.16)**.** *Given $C : [0,1] \to \mathbb{R}_{\ge 0}$. Assume that a denoising process $(Y_t)_{t\in[0,1]}$ is ϕ-entropically stable with rate $C(t)$ throughout $t \in [0,1]$. Then for each given $\theta \in [0,1]$, the process $(Y_t)_{t\in[0,1]}$ will satisfy R-approximate conservation of ϕ-entropy up to time θ where we take*

$$R = \exp\left(\int_0^\theta \frac{C(t)}{1-t} dt\right).$$

Therefore, in order to deduce approximate tensorization of ϕ-entropy of μ, we only need to construct a denoising process Y_t which is ϕ-entropically stable.

From approximate tensorization to functional inequalities for Glauber dynamics

We recall here the fact that approximate tensorization of variance (χ^2 entropy) implies a Poincaré inequality for Glauber dynamics:

▶ **Fact 2.8.** *A distribution μ on $\{0,1\}^n$ satisfies C_1- approximate tensorization of variance if and only if the spectral gap for the Glauber dynamics for μ satisfies $\mathbf{gap}_{GL} \ge \frac{1}{C_1 n}$.*

A similar statement holds for modified log-Sobolev inequalities:

▶ **Fact 2.9** ([10], Fact 3.5)**.** *If a distribution μ on $\{0,1\}^n$ satisfies C_1- approximate tensorization of entropy, then the modified log-Sobolev inequality holds for the Glauber dynamics with constant $\rho_0 \ge \frac{1}{C_1 n}$.*

2.2 A special case: vertex field denoising

An important case of localization schemes is the vertex-tilting field dynamics introduced in [7]. Our edge-tilting localization scheme will also be built on this vertex field version, so we give a quick review here. We consider distributions on $\{0,1\}^n$ and identify a configuration $\sigma \in \{0,1\}^n$ with the subset $\{v \in [n] : \sigma_v = 1\}$.

▶ **Definition 2.10.** *Consider a distribution μ on $\{0,1\}^n$ and a sample $X \sim \mu$. Assign, for each $v \in [n]$, an independent random variable $U_v \sim \mathrm{Uniform}[0,1]$. Then the vertex field denoising process with respect to μ is defined by the following process $(Y_t)_{t\in[0,1]}$:*

$$Y_t := \{v \in [n] : X_v = 1 \wedge U_v \le t\}.$$

For a $\theta \in [0,1]$, the tilted distribution $\theta * \mu$ is defined as

$$(\theta * \mu)(\sigma) \propto \mu(\sigma) \cdot \theta^{|\sigma|}, \quad \forall \sigma \in \{0,1\}^n,$$

where $|\sigma| := |\{v \in [n] : \sigma_v = 1\}|$.

▶ **Lemma 2.11** ([8], Proposition 2.14). *For a distribution μ, let $(Y_t)_{t\in[0,1]}$ be the corresponding vertex field denoising process. Then we have*

$$\mathrm{Law}(Y_1 \mid Y_t) = (1-t) * \mu^{Y_t},$$

such that μ^{Y_t} is the conditional law obtained by setting all variables in Y_t pinned to be 1.

Spectral stability of vertex-field denoising process has the following characterization:

▶ **Lemma 2.12** ([5], Proposition 3.3). *Let $(Y_t)_{t\in[0,1]}$ be a vertex-field denoising process with respect to μ on $\{0,1\}^n$. Then for any fixed $\theta \in [0,1]$, the following two claims are equivalent:*

- *$(Y_t)_{t\in[0,1]}$ is spectrally stable with rate C at time θ;*
- *Given any subset S with $\mathbb{P}[Y_\theta = S] > 0$,*

$$\mathrm{Cov}((1-\theta) * \mu^S) \lesssim C \cdot \mathrm{diag}\{\mathbf{m}((1-\theta) * \mu^S)\},$$

where for a distribution ν, $\mathbf{m}(\nu) := \mathbb{E}_{X\sim\nu}[X]$ is its mean vector.

2.3 Edge-field tilting and its localization schemes

The vertex-field denoising process introduced in the last section changes the strength of the external field of an Ising model but keeps quadratic interaction unchanged. This is not sufficient for the RFIM where we wish to weaken the quadratic interaction but to keep the external field fixed. The recent paper [8] pointed out that our goal can be fulfilled by considering a tilting of the edge-field instead of vertex-field, so we remove edges rather than vertices. Before presenting our concrete constructions of edge-field denoising, we outline the abstract framework of [8].

For a given distribution μ on $\{0,1\}^V$ where V is a ground set, let $\mathcal{A}$ be a collection of events, so that each $A \in \mathcal{A}$ is a subset of $\{0,1\}^V$. We take $X \sim \mu$ and for any event $A \in \mathcal{A}$ define $Z_A = Z_A(X) := \mathbf{1}[X \in A]$. Let $Z = Z(X) := (Z_A)_{A\in\mathcal{A}}$.

We let π denote the joint distribution of (X, Z), so that

$$\pi := \mathrm{Law}(X, Z(X)),$$

so that π is a distribution over $\{0,1\}^{V\cup\mathcal{A}}$. Then by construction, $\mu = \pi_V$ and $\mathrm{Law}(Z) = \pi_{\mathcal{A}}$.

▶ **Definition 2.13** (Event-field dynamics, [8]). *Let $\theta \in [0,1]$. The event-field dynamics for μ with respect to $\mathcal{A}$ and tilt θ is the following defined Markov chain on the state space $\Omega(\mu)$:*

From the current state $X_t \in \Omega(\mu)$, the next state X_{t+1} is generated via

1. *($V \to \mathcal{A}$). Set $Z_t := Z(X_t) = \{A \in \mathcal{A} \mid X_t \in A\}$.*
2. *(Down-$\mathcal{A}$) Then generate a random subset $T \subset Z_t$ where we independently remove each $A \in Z_t$ with probability θ;*
3. *(Up-$\mathcal{A}$) Sample $Z_{t+1} \sim (\theta * \pi_{\mathcal{A}})(\cdot \mid Z_{t+1} \supseteq T)$, where we define*

$$\theta * \pi_{\mathcal{A}}(Z) \propto \pi_A(Z) \cdot \prod_{A\in\mathcal{A}} \theta^{\mathbf{1}(Z_A=1)}, \quad \forall Z \in \{0,1\}^{\mathcal{A}},$$

the distribution obtained from $\pi_{\mathcal{A}}$ by tilting the occurrence of each event A by θ.

4. *Sample $X_{t+1} \sim \pi_V(\cdot \mid Z_{t+1})$.*

The only example that we will use in this paper is the following edge-field dynamics:

▶ **Example 2.14** (Edge-field dynamics)**.** Let $G = (V, E)$ be a graph and μ a distribution on $\{0,1\}^V$. We set

$$\mathcal{A} := \{A_{uv} \mid (u,v) \in E\}, \quad \text{where } A_{uv} := \{\sigma : \sigma_u = \sigma_v = 1\}, \quad \forall (u,v) \in E.$$

Given $T \subseteq \mathcal{A}$ generated in the (Down-$\mathcal{A}$) step, we can compute that

$$\mathbb{P}(X_{t+1} = \sigma \mid T) \propto \mu(\sigma) \cdot \theta^{|\mathcal{A}(\sigma)|} \cdot \mathbf{1}[\mathcal{A}(\sigma) \supseteq T],$$

where $\mathcal{A}(\sigma) := \{A_{uv} \in \mathcal{A} : \sigma \in A_{uv}\}$.

Next we recall the definition of an event-field denoising process from [8]. This is mostly a projection of the vertex-field denoising process on the collection of events $\mathcal{A}$.

▶ **Definition 2.15** (Event-field denoising)**.** *Let μ be a distribution on $\{0,1\}^V$ and consider a collection of events $\mathcal{A}$. Denote by π the joint distribution on $\{0,1\}^{V \cup \mathcal{A}}$ and let $(Y'_t)_{t\in[0,1]}$ be the vertex-field denoising process for the marginal $\pi_{\mathcal{A}}$ with $\pi_{\mathcal{A}} = \mathrm{Law}(Y'_1)$. Then event-field denoising process for μ through $\mathcal{A}$, which we denote by $(Y_t)_{t\in[0,1]}$ is defined via*
1. *Sample $Y_1 \sim \pi_V(\cdot \mid Y'_1)$,*
2. *For $t \in [0,1)$, we set $Y_t = Y'_t$.*

In other words, the process follows the vertex field denoising process applied to the event indicators, and finally at $t = 1$ projects onto the vertices in V via the conditional law $\pi_V(\cdot \mid Y'_1)$.

Since $Y_t = Y'_t$ for all $t \in [0,1)$, the spectral stability of $(Y'_t)_{t\in[0,1]}$ implies the spectral stability of $(Y_t)_{t\in[0,1]}$:

▶ **Lemma 2.16** (ϕ-Entropic stability upgrades, [8], Proposition 4.12)**.** *Let $(Y_t)_{t\in[0,1]}$ and $(Y'_t)_{t\in[0,1]}$ be the two denoising processes in Definition 2.15. Then for any $\theta \in [0,1)$, suppose that $(Y'_t)_{t\in[0,1]}$ is ϕ-entropically stable with rate C at time θ, then $(Y_t)_{t\in[0,1]}$ is also ϕ-entropically stable with rate C at time θ.*

Then we specialize to the edge-field dynamics in Example 2.14 and define its own denoising process:

▶ **Definition 2.17** (Edge-field denoising process)**.** *Let $G = (V, E)$ be a graph and μ a distribution on $\{0,1\}^V$. Recall that we define*

$$\mathcal{A} := \{A_{uv} \mid (u,v) \in E\}, \quad \textit{where } A_{uv} := \{\sigma : \sigma_u = \sigma_v = 1\}, \quad \forall (u,v) \in E.$$

Let $(Y_t)_{t\in[0,1]}$ be the event-field denoising process for μ with respect to this family $\mathcal{A}$. Then the resulting process $(Y_t)_{t\in[0,1]}$ is called the edge-field denoising process for μ on G.

In the following, we give an explicit description of the process Y_t. We shall use a notation $\theta \otimes \mu$ for the distribution obtained by tilting the interactions by a factor $\theta > 0$:

$$\theta \otimes \mu(\sigma) \propto \mu(\sigma) \cdot \theta^{m(\sigma)}, \quad \forall \sigma \in \{0,1\}^V,$$

where we denote by $m(\sigma) := |\{(u,v) \in E : \sigma_u = \sigma_v = 1\}|$ the number of edges whose two endpoints are both equal to 1.

The edge field denoising process $(Y_t)_{t\in[0,1]}$ satisfies that its posterior distribution is the target distribution μ with tilted interaction:

▶ **Proposition 2.18.** *Let $(Y_t)_{t\in[0,1]}$ denote the edge-field denoising process. Then its posterior distribution satisfies*

$$\mathrm{Law}(Y_1 \mid Y_t) = (1-t) \otimes \mu^{Y_t},$$

where we recall that μ^{Y_t} denotes the conditional law of μ conditioning on the event that for every edge $(u,v) \in E$ with $Y_t(u,v) = 1$, then A_{uv} occurs, so that $\sigma_u = \sigma_v = 1$.

Proof. For any $\sigma \in \{0,1\}^V$, denote by $Z(\sigma) \in \{0,1\}^{\mathcal{A}}$ the vector that indicates whether each $A_{uv} \in \mathcal{A}$ occurs in the configuration σ. Take $X \sim \mu$, recall that $\pi := \mathrm{Law}(X, Z(X))$ on $\{0,1\}^{V\cup\mathcal{A}}$.

Let (Y'_t) be the vertex-field denoising process for $\pi_{\mathcal{A}}$. Then for any configuration T,

$$\begin{aligned}
&\mathbb{P}[Y_1 = \sigma \mid Y_t = T] \\
&= \mathbb{P}[Y_1 = \sigma \mid Y'_1 = Z(\sigma)] \cdot \mathbb{P}[Y'_1 = Z(\sigma) \mid Y'_t = T] \\
&\propto \mathbb{P}[Y_1 = \sigma \mid Y'_1 = Z(\sigma)] \cdot (1-t)^{|Z(\sigma)|} \cdot \pi_{\mathcal{A}}(Z(\sigma)) \cdot \mathbf{1}[T \subseteq Z(\sigma)] \\
&\propto (1-t)^{m(\sigma)} \cdot \pi(\sigma, Z(\sigma)) \cdot \mathbf{1}[T \subseteq Z(\sigma)].
\end{aligned}$$

This implies $\mathrm{Law}(Y_1 \mid Y_t) = (1-t) \otimes \mu^{Y_t}$ since σ determines $Z(\sigma)$. ◀

2.4 Spectral stability of edge-field dynamics

Thanks to Lemma 2.16, we only need to prove the spectral stability of the denoising process Y'_t. Motivated by the criterion in Lemma 2.12, we define the following correlation matrix for Y'_t:

▶ **Definition 2.19.** *(Second-order correlation matrix) Let μ be a distribution on $\{0,1\}^V$ and $G = (V,E)$ be a graph. Then for $u, v \in V$, denote by uv the event that*

$$uv := \{\sigma \in \{0,1\}^V \mid \sigma_u = \sigma_v = 1\}.$$

Then we define the second order correlation matrix $\mathrm{Cor}^{(2)}_\mu \in \mathbb{R}^{E\times E}$ via

$$\forall (u,v),(w,z) \in E, \quad \mathrm{Cor}^{(2)}_\mu(uv, wz) := \begin{cases} \mu(wz \mid uv) - \mu(wz), & \text{if } \mu(uv) > 0, \\ 0, & \text{otherwise.} \end{cases}$$

▶ Remark 2.20. We define the correlation matrix $\mathrm{Cor}^{(2)}_\mu$ in this form due to the following fact: let μ be a distribution on $\{0,1\}^V$, then $(\mathrm{diag}(\mathbb{E}(\mu)))^{-1} \cdot \mathrm{Cov}_\mu = \mathrm{Cor}_\mu$.

Although $\mathrm{Cor}^{(2)}_\mu$ is not a symmetric matrix, its eigenvalues are all real.

Then we can translate spectral stability into the following criterion:

▶ **Corollary 2.21.** *(Spectral stability via second-order correlations) Let $(Y_t)_{t\in[0,1]}$ denote the edge-field denoising process for μ on the graph $G = (V,E)$. Then for fixed $\theta \in (0,1)$, assume that for every $\Lambda \subset V$ and any feasible pinning $\tau \in \Omega(\mu_\Lambda)$ we have that*

$$\lambda_{max}(\mathrm{Cor}^{(2)}_{(1-\theta)\otimes\mu^\tau}) \le C,$$

then the process $(Y_t)_{t\in[0,1]}$ is spectrally stable with rate C at time θ.

Proof. This follows from combining Definition 2.15, Lemma 2.16 and Lemma 2.12. Note that a conditioning on the event field events A_{uv} is equivalent to conditioning on the vertex events $\sigma_u = \sigma_v = 1$. ◀

2.5 Approximate tensorization at the end of the flow

We first record how the parameter of RFIM model changes under the edge-field denoising process:

▶ **Fact 2.22.** *Let μ be the RFIM measure μ_G. Then for any $\theta \in (0,1)$ and any pinning τ, the Ising measure $(1-\theta)\otimes\mu^\tau$ has quadratic coefficient $4\beta + \ln(1-\theta)$ (on the coordinate $\{0,1\}^V$ in its Hamiltonian form) on edges of G whose both endpoints are not pinned by τ. In particular, when we take $\theta_* = 1-\exp(-4\beta)$, then $(1-\theta_*)\otimes\mu^\tau$ is a product measure.*

Proof. By Fact 1.6, in the coordinate $\{0,1\}^V$ the quadratic coefficient of $\mu = \mu_G$ in the Hamiltonian is 4β. For each unpinned edge, the denoising process Y_t shrinks the probability by multiplying by $1-\theta$ before normalizing, which means changing 4β to $4\beta + \ln(1-\theta)$ in the Hamiltonian coefficient. ◀

For sake of completeness, we note here that when β is sufficiently small, this model is within the realm of spectral independence where approximate tensorization of variance directly follows. We, however, do not use this estimate as we stop the denoising process at θ_* where the conditional law is the product measure.

▶ **Lemma 2.23.** *Let $G=(V,E)$ be a graph of maximal degree at most Δ, and for each $(u,v)\in E$ we are given constants β_{uv} where $|\beta_{uv}|\le\beta$ for all edges. Fix an arbitrary external field $h=(h_u)_{u\in V}$, an arbitrary subset $A\subset V$ and configuration $\tau\in\{0,1\}^A$. Then whenever $\beta\le\frac{1}{\Delta}$, the measure $\mu_{\beta,h}$ defined on $\{0,1\}^{V\setminus A}$ via*

$$\mu_{\beta,h}(\sigma)\propto\exp\Big(\sum_{(u,v)\in E}\beta_{uv}\sigma_u\sigma_v+\sum_{u\in V\setminus A}h_u\sigma_u\Big),\quad\forall\sigma\in\{0,1\}^V:\sigma|_A=\tau$$

satisfies approximate tensorization of variance and approximate tensorization of entropy with constant 2.

Proof. Let Q denote the quadratic interaction matrix in $\mu_{\beta,h}$ where $Q_{uv}=\beta_{uv}$ if $(u,v)\in E$ and $u\notin A$ and $v\notin A$, and set $Q_{uv}=0$ otherwise. Then $\lambda_{max}(Q)\le\beta\Delta$ and $\lambda_{min}(Q)\ge-\beta\Delta$, where $\lambda_{min}(Q),\lambda_{max}(Q)$ are the smallest and largest eigenvalues of Q. Then via Fact 1.6, we can rewrite $\mu_{\beta,h}$ as an Ising model on $\{-1,1\}^V\setminus A$ where the quadratic interaction matrix is $\frac{1}{4}Q$. Then by [2], Theorem 12, $\mu_{\beta,h}$ satisfies approximate tensorization of variance and entropy with constant $\frac{1}{1-(\lambda_{max}(\frac{1}{4}Q)-\lambda_{min}(\frac{1}{4}Q))}\le 2$. ◀

Now we summarize what we have gotten so far, and what remains to be proven in the remaining sections. We will apply the edge field denoising process Y_t to the RFIM μ_G, where we take

$$\theta_* = 1-\exp(-4\beta) \tag{2.1}$$

as a threshold value. This value of θ_* is fixed throughout the paper. We need to upper bound the integral in R in Lemma 2.7 up to time θ_*, and then tensorization of variance/entropy is free for $\mathrm{Law}(Y_1 \mid Y_{\theta_*})$ since the latter is a product measure. For the integral in R, by Proposition 2.18, it suffices to control the second correlation matrix for $(1-\theta)\otimes\mu^\tau$ for all $0\le\theta\le\theta_*$ and all pinning τ. In the next section, we will focus on the operator norm control of that matrix.

3 The spectral norm of correlation matrix with a random field

In this section, we make the following assumption on the external field of RFIM, which is more general than Assumption 1.1 as we only require $|h_v|$ to be independent, but the signs of h_v are not necessarily independent.

▶ **Assumption 3.1.** *Let $p_0 \in (0,1)$ be given such that $p_0(\Delta-1)<1$, let $K = K(p_0) > 0$ be a constant such that*

$$\rho = \rho(\beta, \Delta, K) := \frac{e^{\Delta\beta - K}}{e^{\Delta\beta - K} + e^{-\Delta\beta + K}}, \tag{3.1}$$

is less than $\frac{p_0}{4}$. Suppose that the RFIM μ_G with external field h is such that $(|h_u|)_{u\in V}$ are i.i.d. and such that

$$\mathbb{P}_h(|h_u| \le K) < \frac{p_0}{2}.$$

▶ **Proposition 3.2.** *Let $(u,v) \in E$ be any edge of the graph $G=(V,E)$ of maximal degree Δ. Assume that Assumption 3.1 holds for the given $p_0 \in (0,1)$, then we have, for any $m \ge 1$,*

$$\begin{aligned} &\mathbb{P}\left(\sup_{\theta\in[0,\theta_*],\Lambda,\tau_0\in\Omega(\Lambda)} \sum_{(wz)\in E} |\operatorname{Cor}^{(2)}_{(1-\theta)\otimes\mu^{\tau_0}}(uv,wz)| \ge \Delta m \right) \\ &\le \frac{2}{(1-e^{-\alpha_*})(1-e^{-2\alpha_*})} e^{2\gamma_*} e^{-\alpha_* m}, \end{aligned} \tag{3.2}$$

where $\gamma_* := \ln \frac{1-p_0}{p_0(\Delta-2)}$ and $\alpha_* = \frac{1}{2}\log \frac{(\Delta-2)^{\Delta-2}}{p_0(\Delta-1)^{\Delta-1}(1-p_0)^{\Delta-2}}$.

Proof. For fixed edge (u,v) and the Ising measure $\nu := (1-\theta)\otimes\mu^{\tau_0}$, we construct a coupling of two spins $\sigma^1 \sim \nu$ and $\sigma^2 \sim \nu(\cdot \mid uv)$, where in σ^2 the vertices u, v are both fixed to be 1 and then we reveal σ^1 at u,v and then reveal both σ^1, σ^2 at vertices in $V\setminus\{u,v\}$. Assign uniform random variables $U_x, x\in V$ at each vertex of V, and generate $(\sigma^1)_u$ by the given probability law of ν, then generate $(\sigma^1)_v$ by the given probability law of ν conditioning on σ^1_u. Then we iteratively reveal the vertices of distance 1 to (u,v) in both configurations σ^1,σ^2 in any given sequence, and update the value of σ^1 and σ^2 with respect to the probability laws ν and $\nu(\cdot\mid uv)$, conditioning on the vertices already revealed. Use the same uniform random variable to update each vertex. Note that when we update both σ^1 and σ^2 at $w\neq u,v$, then when $|h_w|>K$, then the probability that the updated spins in σ^1,σ^2 have the sign of h_w is at least $1-\frac{p_0}{4}$, by Assumption 3.1. (the probability that σ^i_w has the same sign as h_w can be computed in the original coordinate $\{-1,1\}^V$ where the expression $\rho(\beta,\Delta,K)$ in Assumption 3.1 is more apparent) Therefore, the subset of V where σ^1 and σ^2 disagree is controlled by the connected component of (u,v) in the percolation process

$$\{|h_x| \le K\} \cup \{\min(U_x, 1-U_x) \le \frac{p_0}{4}\} \tag{3.3}$$

at each $x\in V$, since outside this connected component, σ^1 and σ^2 are updated via the identity coupling. By definition of total variation distance, we have

$$|\operatorname{Cor}^{(2)}_\nu(uv,wz)| \le \mathbb{P}(\sigma^1_{wz} \neq \sigma^2_{wz}) \le \mathbb{P}(\sigma^1_w \neq \sigma^2_w) + \mathbb{P}(\sigma^1_z \neq \sigma^2_z)$$

for our specific coupling (here $\sigma^1_{wz}\neq\sigma^2_{wz}$ means $\sigma^1_w\neq\sigma^2_w$ or $\sigma^1_z\neq\sigma^2_z$), and thus since each vertex is adjacent to at most Δ edges, we multiply by Δ and write

$$\sum_{(wz)\in E} |\operatorname{Cor}^{(2)}_\nu(uv,wz)| \le \Delta\cdot \mathbb{E}_U|\text{connected component of } (u,v) \text{ in } \mathrm{Ber}(p_0) \text{ percolation}|,$$

where the left hand side is a function of h only and the right hand side takes expectation with respect to the uniform measure $(U_x, x \in V)$ in defining the coupling, and the percolation is defined in (3.3). Let $\mathcal{C}_{h,U}(u,v)$ denote the component of (u,v) in this percolation, and denote by $S_{uv}(h) := \frac{1}{\Delta}\sum_{(wz)\in E} |\operatorname{Cor}^{(2)}_{\nu}(uv,wz)|$ (This notation has implicit dependence on θ, Λ, τ_0.) Then for any $s > 0$,

$$\mathbb{P}_h(S_{uv}(h) \geq m) \leq e^{-sm}\mathbb{E}_h \exp(s\mathbb{E}_U|\mathcal{C}_{h,U}|) \leq e^{-sm}\mathbb{E}_{h,U}\exp(s|\mathcal{C}_{h,U}|),$$

applying Jensen in the last step.

Moreover, for any $\theta \in [0,\theta_*]$, any Λ and any boundary condition τ_0, we use the same coupling for ν and $\nu(\cdot \mid uv)$ via the same uniform distribution, and thus the discrepancy region is simultaneously controlled by the same Bernoulli percolation (3.3). That is,

$$\mathbb{P}_h(\sup_{\theta\in[0,\theta_*],\Lambda,\tau_0} S_{uv}(h) \geq m) \leq e^{-sm}\mathbb{E}_{h,U}\exp(s|\mathcal{C}_{h,U}|).$$

In a subcritical Bernoulli percolation, the size of the connected component has exponential tails, see equation (3.5). Taking $s = \alpha_*$ defined below and using equation (3.5), we deduce that (3.2) holds and thus the Proposition is proven.

The rest of the proof is the derivation of (3.5) for the exponential moments for the size of the connected component. We consider a Bernoulli (p_0)-percolation on G whose law is denoted by $\mathbb{P}$, and we use $\mathcal{C}(u,v)$ denote the connected cluster in this percolation where (u,v) are forced open. Then we explore the open cluster of the edge (u,v). Since the maximal degree is at most Δ and (u,v) are forced open, the connected cluster is stochastically bounded by a Galton-Watson forest with two initial particles and offspring distribution $\mathrm{Bin}(\Delta-1,p_0)$, see for example [17]. By the Otter-Dwass formula [13] (and see [23] for the $r=2$ version), if we let $T^{(2)}$ denote the total progeny of the forest, then

$$\mathbb{P}(T^{(2)} = x) = \frac{2}{x}\mathbb{P}(\mathrm{Bin}((\Delta-1)x, p_0) = x-2), \quad \forall x \in \mathbb{N}_+.$$

By Chernoff, whenever $p_0(\Delta-1) < 1$, we have

$$\mathbb{P}(T^{(2)} = x) \leq \frac{2}{x}\left(\frac{1-p_0}{p_0(\Delta-2)}\right)^2 e^{-\xi_* x},$$

where we take $\xi_* = \log\frac{(\Delta-2)^{\Delta-2}}{p_0(\Delta-1)^{\Delta-1}(1-p_0)^{\Delta-2}}$. Then we sum up over integers $x \geq m$ and get

$$\mathbb{P}(|\mathcal{C}(u,v)| \geq m \mid (u,v) \text{ open }) \leq \frac{2}{1-e^{-\xi_*}}\left(\frac{1-p_0}{p_0(\Delta-2)}\right)^2 e^{-\xi_* m}. \tag{3.4}$$

Then we can compute the exponential moment of $\mathcal{C}(u,v)$: take $\alpha_* = \frac{1}{2}\xi_*$, then

$$\mathbb{E}[\exp(\alpha_*|\mathcal{C}(u,v)|) \mid (u,v) \text{ open }] \leq \frac{2}{(1-e^{-2\alpha_*})(1-e^{-\alpha_*})}\left(\frac{1-p_0}{p_0(\Delta-2)}\right)^2. \tag{3.5}$$

◀

Via exactly the same argument, we have the following estimate:

▶ **Proposition 3.3.** *Let $(w,z) \in E$ be any edge of the graph $G = (V,E)$ of maximal degree Δ. Assume that Assumption 3.1 holds for the given $p_0 \in (0,1)$, then we have, for any $m \geq 1$,*

$$\begin{aligned}&\mathbb{P}\left(\sup_{\theta\in[0,\theta_*],\Lambda,\tau_0\in\Omega(\Lambda)}\sum_{(uv)\in E}|\operatorname{Cor}^{(2)}_{(1-\theta)\otimes\mu^{\tau_0}}(uv,wz)| \geq 2\Delta m\right)\\ &\leq \frac{2}{(1-e^{-\alpha_*})(1-e^{-2\alpha_*})}e^{2\gamma_*}e^{-\alpha_* m}.\end{aligned} \tag{3.6}$$

Proof. Consider the same Bernoulli percolation process as in (3.3), and for each $\theta \in (0, \theta_*)$ and pinning τ_0, use the same uniform coupling to generate $\sigma^1 \sim \nu := (1-\theta) \otimes \mu^{\tau_0}$ and $\sigma^2 \sim \nu(\cdot \mid wz)$. By the same argument in the proof of Proposition 3.2, we have

$$\begin{aligned}
&\sum_{(uv)\in E} |\operatorname{Cor}^{(2)}_{\nu}(uv, wz)| \\
&\le 2\mathbb{E}_U \sum_{(uv)\in E} \mathbf{1}((wz) \text{ is in the connected component of } (uv) \text{ in this percolation}) \\
&= 2\mathbb{E}_U \sum_{(uv)\in E} \mathbf{1}((uv) \text{ is in the connected component of } (wz) \text{ in this percolation}) \\
&\le 2\Delta \cdot \mathbb{E}_U |\text{connected component of } (wz) \text{ in } \mathrm{Ber}(p_0) \text{ percolation}|.
\end{aligned}$$

Denote by $S^{\mathrm{col}}_{wz}(h) := \frac{1}{2\Delta} \sum_{(uv)\in E} |\operatorname{Cor}^{(2)}_{\nu}(uv, wz)|$. Then

$$S^{\mathrm{col}}_{wz}(h) \le \mathbb{E}_U |\mathcal{C}_{h,U}(w, z)|.$$

Then the same Chernoff bound and exponential moment applies, concluding the proof. ◀

For a square matrix $A \in \mathbb{R}^{n\times n}$, its operator norm satisfies the following interpolation inequality

$$\|A\| \le \sqrt{\left(\sup_{i\in[n]} \sum_{j=1}^{n} |a_{ij}|\right)\left(\sup_{j\in[n]} \sum_{i=1}^{n} |a_{ij}|\right)}. \tag{3.7}$$

Taking a union bound in Proposition 3.2, we obtain an operator norm upper bound for all Ising models $(1-\theta) \otimes \mu^{\tau_0}$:

▶ **Proposition 3.4.** *Under the same assumption in Proposition 3.2, with probability at least* $1 - O_{\beta,\Delta,p_0}(\frac{1}{|V|})$ *over the randomness in the external field* h*,*

$$\sup_{\theta\in[0,\theta_*],\Lambda,\tau_0\in\Omega(\Lambda)} \| \operatorname{Cor}^{(2)}_{(1-\theta)\otimes\mu^{\tau_0}} \|_{op} \le \frac{4\Delta \ln |V|}{\alpha_*}.$$

Proof. We take $m = \frac{2\ln|V|}{\alpha_*}$ in the estimate (3.2) for each edge (u, v) and take the same value of m in the estimate (3.6) for each edge (w, z). Then with probability at least $1 - 2|V|\Delta \cdot \frac{2}{(1-e^{-\alpha_*})(1-e^{-2\alpha_*})} e^{2\gamma_*} \frac{1}{|V|^2}$, we have

$$\sup_{(uv)\in E} \sum_{(wz)\in E} |\operatorname{Cor}^{(2)}_{(1-\theta)\otimes\mu^{\tau_0}}(uv, wz)|, \ \sup_{(wz)\in E} \sum_{(uv)\in E} |\operatorname{Cor}^{(2)}_{(1-\theta)\otimes\mu^{\tau_0}}(uv, wz)| \le 2\Delta m.$$

Taking these two estimates into (3.7) for $A = \operatorname{Cor}^{(2)}_{(1-\theta)\otimes\mu^{\tau_0}}$ completes the proof. ◀

▶ **Corollary 3.5.** *Under the same assumption in Proposition 3.2, with* $\mathbb{P}_h$*-probability at least* $1 - O_{\beta,\Delta,p_0}(\frac{1}{|V|})$*, the edge field denoising process* $(Y_t)_{t\in[0,1]}$ *satisfies* R*- approximate conservation of variance up to time* θ_* *with constant*

$$R \le \exp\left(\frac{16\beta\Delta \ln |V|}{\alpha_*}\right).$$

Moreover, the RFIM measure μ_G *satisfies approximate tensorization of variance with constant* R*.*

Proof. Note that $\int_0^{\theta_*} \frac{dt}{1-t} = 4\beta$. The claim on (Y_t) follows directly from Proposition 3.4, Lemma 2.7 and Corollary 2.21. The R-approximate tensorization of variance of μ_G then follows from Lemma 2.5, since the posterior $\mathrm{Law}(Y_1 \mid Y_{\theta_*})$ is a product measure. ◀

Now we can conclude the proof of Theorem 1.2:

Proof of Theorem 1.2. The claim on Poincaré constant follows from Fact 2.8 combined with Corollary 3.5. The claim on mixing times follows from Fact 1.5. ◀

References

1 Michael Aizenman and Jan Wehr. Rounding effects of quenched randomness on first-order phase transitions. *Communications in mathematical physics*, 130(3):489–528, 1990.

2 Nima Anari, Vishesh Jain, Frederic Koehler, Huy Tuan Pham, and Thuy-Duong Vuong. Entropic independence i: Modified log-sobolev inequalities for fractionally log-concave distributions and high-temperature ising models. *arXiv preprint*, 2021. arXiv:2106.04105.

3 Jean Bricmont and Antti Kupiainen. Phase transition in the 3d random field ising model. *Communications in mathematical physics*, 116(4):539–572, 1988.

4 Jin-Yi Cai, Andreas Galanis, Leslie Ann Goldberg, Heng Guo, Mark Jerrum, Daniel Štefankovič, and Eric Vigoda. # bis-hardness for 2-spin systems on bipartite bounded degree graphs in the tree non-uniqueness region. *Journal of Computer and System Sciences*, 82(5):690–711, 2016. doi:10.1016/J.JCSS.2015.11.009.

5 Xiaoyu Chen, Zejia Chen, Zongchen Chen, Yitong Yin, and Xinyuan Zhang. Rapid mixing on random regular graphs beyond uniqueness. *arXiv preprint*, 2025. doi:10.48550/arXiv.2504.03406.

6 Xiaoyu Chen, Zongchen Chen, Yitong Yin, and Xinyuan Zhang. Rapid mixing at the uniqueness threshold. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 879–890, 2025. doi:10.1145/3717823.3718260.

7 Xiaoyu Chen, Weiming Feng, Yitong Yin, and Xinyuan Zhang. Rapid mixing of glauber dynamics via spectral independence for all degrees. *SIAM Journal on Computing*, pages FOCS21–224, 2024.

8 Xiaoyu Chen, Zhe Ju, Tianshun Miao, Yitong Yin, and Xinyuan Zhang. Edge-tilting field dynamics: Rapid mixing at the uniqueness threshold and optimal mixing for swendsen-wang dynamics. *arXiv preprint*, 2026. doi:10.48550/arXiv.2604.10525.

9 Yuansi Chen and Ronen Eldan. Localization schemes: A framework for proving mixing bounds for markov chains. In *2022 IEEE 63rd Annual symposium on foundations of computer science (FOCS)*, pages 110–122. IEEE, 2022.

10 Zongchen Chen, Kuikui Liu, and Eric Vigoda. Optimal mixing of glauber dynamics: Entropy factorization via high-dimensional expansion. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1537–1550, 2021. doi:10.1145/3406325.3451035.

11 Jian Ding, Yu Liu, and Aoteng Xia. Long range order for three-dimensional random field ising model throughout the entire low temperature regime. *Inventiones mathematicae*, 238(1):247–281, 2024.

12 Jian Ding and Jiaming Xia. Exponential decay of correlations in the two-dimensional random field ising model. *Inventiones mathematicae*, 224(3):999–1045, 2021.

13 Meyer Dwass. The total progeny in a branching process and a related random walk. *Journal of Applied Probability*, 6:682–686, 1969.

14 Ahmed El Alaoui, Ronen Eldan, Reza Gheissari, and Arianna Piana. Fast relaxation of the random field ising dynamics. *The Annals of Probability*, 54(1):99–136, 2026.

15 Ronen Eldan, Frederic Koehler, and Ofer Zeitouni. A spectral condition for spectral gap: fast mixing in high-temperature ising models. *Probability theory and related fields*, 182(3):1035–1051, 2022.

16 Roy J Glauber. Time-dependent statistics of the ising model. *Journal of mathematical physics*, 4(2):294–307, 1963.

17 Geoffrey Grimmett. *Percolation*. Springer, 2 edition, 1999.

18 Tyler Helmuth, Holden Lee, Will Perkins, Mohan Ravichandran, and Qiang Wu. Approximation algorithms for the random field ising model. *SIAM Journal on Discrete Mathematics*, 37(3):1610–1629, 2023. doi:10.1137/21M1467389.

19 John Z Imbrie. The ground state of the three-dimensional random-field ising model. *Communications in mathematical physics*, 98(2):145–176, 1985.

20 Yoseph Imry and Shang-keng Ma. Random-field instability of the ordered state of continuous symmetry. *Physical Review Letters*, 35(21):1399, 1975.

21 David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, Providence, RI, 2 edition, 2017.

22 Fabio Martinelli. Lectures on Glauber dynamics for discrete spin models. In *Lectures on Probability Theory and Statistics*, volume 1717 of *Lecture Notes in Mathematics*, pages 93–191. Springer, Berlin, 1999. doi:10.1007/978-3-540-48115-7_2.

23 Jim Pitman. *Combinatorial Stochastic Processes*, volume 1875 of *Lecture Notes in Mathematics*. Springer, 2006.

24 Dror Weitz. Counting independent sets up to the tree threshold. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 140–149, 2006. doi:10.1145/1132516.1132538.

Markov Chains with Rewinding

Amir Azarmehr
Northeastern University, Boston, MA, USA

Soheil Behnezhad
Northeastern University, Boston, MA, USA

Alma Ghafari
Northeastern University, Boston, MA, USA

Madhu Sudan
Harvard University, Cambridge, MA, USA

Abstract

Motivated by techniques developed in recent progress on lower bounds for sublinear time algorithms (Behnezhad, Roghani and Rubinstein, STOC 2023, FOCS 2023, and STOC 2024) we introduce and study a new class of *randomized algorithmic* processes that we call "Markov Chains with Rewinding". In this setting an agent/*algorithm* interacts with a (partially observable) Markovian *random* evolution by periodically/strategically rewinding the Markov chain to previous states. Depending on the application this may lead the evolution to desired states faster, or allow the agent to efficiently learn or test properties of the underlying Markov chain that may be infeasible or inefficient with passive observation.

We study the task of identifying the initial state in a given partially observable Markov chain. Analysis of this question in specific Markov chains is the central ingredient in the above cited works and we aim to systematize the analysis in our work. Our first result is that any pair of states distinguishable with any rewinding strategy can also be distinguished with a *non-adaptive* rewinding strategy (i.e., one whose rewinding choices are predetermined before observing any outcomes of the chain). Therefore, while rewinding strategies can be shown to be strictly more powerful than passive strategies (i.e., those that do not rewind back to previous states), adaptivity does *not* give additional power to a rewinding strategy in the absence of efficiency considerations.

The difference becomes apparent however when we introduce a natural efficiency measure, namely the *query complexity* (i.e., the number of observations they need to identify distinguishable states). Our second main contribution is to quantify this efficiency gap. We present a non-adaptive rewinding strategy whose query complexity is within a polynomial of that of the optimal (adaptive) strategy, and show that such a polynomial loss is necessary in general.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Partially observable Markov chains, rewinding algorithm, sublinear algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.43

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2602.16028

Funding Amir Azarmehr, Soheil Behnezhad, and Alma Ghafari were supported in part by NSF CAREER Award CCF-2442812 and a Google Research Award. Madhu Sudan was supported in part by a Simons Investigator Award, NSF Award CCF 2152413 and AFOSR award FA9550-25-1-0112.

1 Introduction

In this work, we formally define and study a new class of processes involving interaction between randomness and algorithms. In these processes, that we call *Markov chains with rewinding*, a partially observable Markov chain interacts with an algorithmic agent that has

© Amir Azarmehr, Soheil Behnezhad, Alma Ghafari, and Madhu Sudan;
licensed under Creative Commons License CC-BY 4.0

the ability, at any point in time, to rewind the Markov chain to a previous point of time. We study the problem of *identifying the initial state* in this model, where the goal is for the algorithm to use the observations and the power of rewinding to identify a given (hidden) initial state.

Partially observable Markov chains form a well-established element of the toolkit in computing and arise in both the design and analysis of algorithms. Rewinding of Markov chains captures a natural ability of algorithms that work with Markov chains, and as a result, they have occurred implicitly in a variety of settings (an overview is given in Section 1.2).

This is highlighted centrally in a class of applications of Markov chains with rewinding to *sublinear time algorithms*. For the maximum matching problem, [8, 7, 9] prove strong lower bounds on the query complexity, and hence running time, of sublinear time algorithms by using Markov chains with rewinding to model sublinear algorithms, and then analyzing these chains. Recent works on non-adaptive sublinear lower bounds follow similar paradigms, while not explicitly defining a Markov chain [5, 25]. Additionally, lower bounds for other graph problems in the sublinear-time model, such as edge orientation [22] and acyclicity [11], can also be viewed as instances of this general model.

Despite the prevalence of this phenomenon, rewinding does not appear to have been studied formally in the literature and in particular tools to analyze the power (and limitations) of Markov chains with rewinding have been ad-hoc. Our work is motivated principally by these and aims to build a more systematic toolkit for such analysis.

Markov chains with rewinding. We start with an example to illustrate the problem. Consider the partially observable Markov chain depicted below, where the only observable variable is the color of the current state (i.e., whether it is in $\{a, a', b, b'\}$ or $\{s\}$). The initial state is promised to be either a or a', and our goal is to identify this initial state.

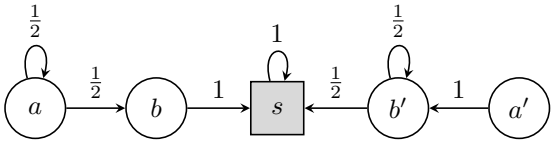


Figure 1.1 Rewinding is stronger than passive observation.

First, note that "passive observations" cannot distinguish a from a': Whether we start from a or a', it takes exactly Geometric$(1/2) + 1$ steps until we reach s and remain there forever. But suppose we are allowed to *rewind* back to any previous state at any time. In this case, we can run the Markov chain for one step to reach state X, and then run D independent steps from X for a sufficiently large D. The idea is that if $X = b'$ (which implies the initial state is a'), then it will most likely have a mix of s and not s next steps, whereas if $X \in \{a, b\}$ (implying that the initial state is a) all next steps will be the same. The following figure illustrates this.

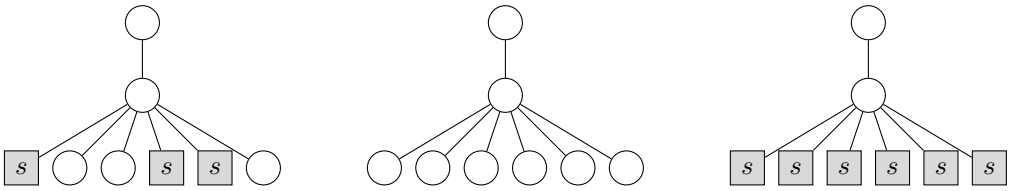


Figure 1.2 Examples of states generated by repeatedly rewinding to the second step.

We now present a more general description of the model, deferring its formalization to Section 3. The input consists of a partially observable Markov chain M, two candidate initial states a and a', and a hidden initial state $X_0 \in \{a, a'\}$. The algorithm interacts with the Markov chain through the hidden initial state X_0. It can either draw states according to the transition probabilities, or rewind the state to an earlier point in time. The drawn states $\{X_t\}_t$ are hidden from the algorithm, but observations $\{Z_t\}_t$ are available, where $Z_t = O(X_t)$. Here O is a fixed observation function mapping each state to its observation, so that $O(X_t)$ is the observation revealed at time t. Eventually, the algorithm must decide whether the initial state X_0 is a or a', based on the observations. We study the runtime and the number of queries (i.e. drawn states) required to do so. Other than the motivation from sublinear-algorithm lower bounds, rewinding formalizes a natural approach to interacting with Markov chains; variations and special cases have been considered repeatedly in previous work (see Section 1.2).

1.1 Our Contributions

Our main focus in this work is on the following fundamental problem:

> *Given a partially observable Markov chain, can a nearly-optimal rewinding strategy be computed efficiently?*

To better understand this question, we first need to formalize the model. We present the formalization in Section 3; We define *rewinding strategies* and *state identification algorithms*. Rewinding strategy refers to the choice of rewinding times throughout the process. A state identification algorithm computes the rewinding strategy through which it interacts with the partially observable Markov chain, and ultimately identifies the initial state based on the observations. We study and compare the power of *adaptive* and *non-adaptive* rewinding strategies for this task, where a non-adaptive strategy determines how to rewind independently of the observations. These lead us to our key complexity measures $\mathsf{QC}_M(a, b)$ (resp. $\mathsf{NAQC}_M(a, b)$) for the adaptive (resp. non-adaptive) *query complexity* of distinguishing state a from b in chain M, i.e. the minimum number of observations required by a rewinding strategy to distinguish the two states. Query complexity provides a benchmark for measuring the efficiency of an algorithm, and a means of proving lower bounds for its runtime.

Crucially, the algorithm is given the full description of the chain $M = (\Omega, P, O)$, here Ω is the (finite) state space of the Markov chain, $\Omega \times \Omega \to [0, 1]$ is its transition probability matrix, and O is the observation function introduced above. What is hidden is only the realized trajectory $\{X_t\}_t$ and the initial state.

We show that there is indeed a polynomial-time algorithm for the question stated at the beginning of this section. Formally, we prove the following theorem.

▶ **Theorem 1.1.** *Given a partially observable Markov chain $M = (\Omega, P, O)$, there exists a non-adaptive algorithm and a constant $c = c(|\Omega|)$ that can distinguish between any two states $a, b \in \Omega$ in time*

$$c \cdot \mathsf{QC}_M(a,b)^{O(|\Omega|)}.$$

Note that we are using the query complexity as our measure of efficiency. That is, we devise a state identification algorithm that chooses the rewinding states in time $\text{poly}(\mathsf{QC}_M(a,b))$ and generates at most $\text{poly}(\mathsf{QC}_M(a,b))$ states. In the statement above, we let M be a Markov chain on state space Ω with a special *sink* state. The only observable signal available to the rewinding algorithm is whether the chain is in the sink state or not (similar to the example in Section 1). We call such Markov chains *canonical* and show that identifying the initial state in canonical Markov chains is essentially as hard as general partially observable Markov chains (see Section 6).

Observe that if non-adaptive algorithms cannot distinguish two states a and b of a Markov chain M, then Theorem 1.1 implies that no adaptive algorithm can do so either. Therefore, non-adaptive algorithms are as powerful as adaptive ones in terms of which pairs of states they can distinguish.

Furthermore, formalized as Theorem 1.2, we show that the power of $\mathsf{QC}_M(a,b)$ in Theorem 1.1 is tight, up to a constant, for the runtime of non-adaptive algorithms. More specifically, we present a partially observable Markov chain M and states a and b, where a polynomial gap exists between the adaptive and non-adaptive query complexity of distinguishing a and b.

▶ **Theorem 1.2.** *There is a choice of M and pair $a, b \in \Omega$ such that for some $c = c(|\Omega|)$,*

$$\mathsf{NAQC}_M(a,b) \geq c \cdot \mathsf{QC}_M(a,b)^{(1-o(1))|\Omega|}.$$

This holds for infinitely many choices of $|\Omega|$ and query complexities.

We remark that the adaptive rewinding strategy presented for this gap can be easily turned into an algorithm with the same run time. Therefore, the polynomial gap is also present between the optimal adaptive and non-adaptive run times of state identification algorithms.

We leave open the question of whether the number of states in the Markov chain can be removed from the run-time exponent. Any solution would require new techniques and necessarily rely on adaptivity, as remarked above. Formally, we ask:

▶ **Open Problem 1.** *Given a partially observable Markov chain $M = (\Omega, P, O)$, candidate initial states $a, b \in \Omega$, and hidden initial state $X_0 \in \{a, b\}$, is there an adaptive state identification algorithm that successfully identifies the initial state in time $\mathsf{QC}_M(a,b)^C$ where constant C is independent of the number of states?*

We believe it is also interesting to compare non-adaptive algorithms against the best non-adaptive benchmark for future work. In particular, we ask:

▶ **Open Problem 2.** *Given a partially observable Markov chain $M = (\Omega, P, O)$, candidate initial states $a, b \in \Omega$, and hidden initial state $X_0 \in \{a, b\}$, is there a non-adaptive state identification algorithm that successfully identifies the initial state in time $\mathsf{NAQC}_M(a,b)^C$ where constant C is independent of the number of states?*

1.2 Related Work

We overview settings where partially observable Markov chains with rewinding have occurred naturally. A highly prevalent instance is the folklore method of converting expected *run times of algorithms* into high probability bounds by restarting the algorithm when it does not halt promptly. This is a special case of rewinding where the underlying state of the algorithm is captured by a Markov chain, and the observation is limited to knowing whether the algorithm has halted or not.

Restarting/rewinding can also help boost the success of algorithms as exemplified in the famed randomized algorithm for 3SAT due to Schöning [24]. In the case of random walks on multi-dimensional lattices, [21] showed that the expected hitting time of the origin is finite when the algorithm is allowed to restart the walk back from the starting point. More general forms of rewinding lead to elegant new approaches to *optimization* and *random sampling* as in the "Go-with-the-winners" algorithm of Aldous and Vazirani [1]. Selecting which state to continue the process from is also studied by [18], where the algorithm decides which Markov chain to advance, in order to minimize the expected time that one of them reaches the target state. In *cryptography*, rewinding arguments play a celebrated role in proving the security of protocols starting with the seminal work of Goldwasser, Micali, and Rackoff [20]. Rewinding is also a central element in defining strong notions of security as in the notion of "resettable zero knowledge" due to Canetti, Goldreich, Goldwasser, and Micali [13].

A related line of work studies other forms of controlled Markov chains. For instance, Azar, Broder, Karlin, Linial, and Phillips [4] introduced biased random walks, in which a controller may, with small probability, override the random transition to steer the walk, and gave tight bounds and polynomial-time algorithms for the optimal control strategy maximizing a long-term benefit.

1.3 Connection to Sublinear-Time Graph Algorithms

To further motivate our model, we overview its connection to sublinear time graph algorithms (or more precisely, lower bounds). Here, we provide a high-level description of this connection in general. Later in Appendix A, we present an explicit connection to the lower bound for testing acyclicity. That is, we show how the lower bound is captured by Markov chains with rewinding.

Let us first briefly overview the model of sublinear-time algorithms for graphs. We are given a graph $G = (V, E)$ to which we have adjacency list query access.[1] Each query can specify a vertex v and an integer $i \geq 1$. The answer to such a query is the i-th neighbor of vertex v stored in an arbitrarily ordered list, or $\perp$ if $\deg(v) < i$. Now, the main question of interest, is the number of such queries needed to estimate a property of the graph. For example, the works of [8, 7, 9] focus on proving lower bounds on the query complexity of estimating the size of maximum matching in the graph (see also [6, 26, 23, 12] for some algorithms). Many other problems can, and have been, studied in this setting, including graph coloring [2], correlation clustering [3], metric properties such as minimum spanning trees [17, 14], Steiner trees [15], TSP [16, 10], and many others.

Remarkably, for many graph properties, we have had *unconditional* query lower bounds that can be matched algorithmically with sublinear time algorithms. However, these lower bound arguments are often ad hoc, and there is a lack of systematic tools to prove them. Our hope is that our model and techniques can serve as a first step towards developing such generic lower bound tools. Let us expand on this connection a bit further.

[1] Another common access model is the adjacency matrix model.

Typically, sublinear time lower bounds are proved by specifying two distributions $\mathcal{D}_{YES}$ and $\mathcal{D}_{NO}$ of graphs, where graphs drawn from $\mathcal{D}_{YES}$ have the desired property and those drawn from $\mathcal{D}_{NO}$ do not. The lower bound then follows by showing that distinguishing these distributions requires many queries to the graph. In most cases, these distributions are constructed by having a few groups $A_1, A_2, \ldots, A_k$ that each vertex may belong to and is kept hidden from the algorithm. The distribution of edges between these groups differs in $\mathcal{D}_{YES}$ and $\mathcal{D}_{NO}$, and thus the task of distinguishing these distributions reduces to identifying these groups. The only useful signal that the algorithm receives, and helps in identifying the groups, is the vertex degrees. This is precisely how the hard instances of [8, 7, 9] for maximum matchings are constructed. Lower bounds for other graph problems in the sublinear-time model, such as edge orientation [22] and acyclicity [11], can also be viewed as instances of this general framework.

Now note that each group A_i can be viewed as a different state of the Markov chain. We let p_{ij} (i.e., the transition probability from A_i to A_j) to be the fraction of edges of group A_i that go to A_j, representing the fact that a random adjacency list neighbor of an A_i node belongs to A_j with probability p_{ij}. We construct these states separately for groups in $\mathcal{D}_{YES}$ and $\mathcal{D}_{NO}$. Then, assuming that the algorithm cannot find cycles (which holds in many lower bounds such as [8, 7, 19, 11]) the task of distinguishing the two distributions reduces to distinguishing if the initial state is among the YES states or the NO states of the Markov chain. Note, in particular, that because the sublinear time algorithm is not constrained to follow a single path in the graph and can adaptively make adjacency list queries to any previously visited vertex, this corresponds to a rewinding strategy on the Markov chain. Therefore, the study of the power and limitations of such rewinding strategies for general Markov chains closely relates to the query complexity of sublinear-time graph algorithms, and can serve as a rich toolkit for proving lower bounds against them.

Paper Arrangement

More examples of Markov chains with rewinding are presented in Section 2. An overview of our techniques appears in Section 2.3. The model is formally defined in Section 3. We examine state partitions, a key tool for our algorithm, in Section 3.1. Proofs of Theorems 1.1 and 1.2 are sketched in Sections 4 and 5 respectively. Finally, in Section 6, we prove the generality of canonical Markov chains. For the full proofs refer to the following version [5].

2 Motivating Examples & Technical Overview

We start this section by presenting examples of rewindable Markov chains along with optimal strategies to distinguish between two candidate initial states. These examples further clarify the model and demonstrate how rewinding strategies can be *non-trivially* effective, and thus hard to prove lower bounds against. After these examples, we outline the techniques used in our algorithms and analyses.

2.1 Example 1: The non-trivial power of (non-adaptive) rewinding

Our first example involves a simple Markov chain on 5 states, inspired by the hard distribution of [8] for graph matchings. The chain is parameterized by an integer d, and its transition probabilities are multiples of $1/d$ as illustrated by the arrows in the figure. There is a special sink state s, and the algorithm can only observe whether a state is sink or non-sink. The goal is to distinguish whether the initial state is a or a'. We show first how the natural strategy to distinguish two candidate initial states a and a' leads to query complexity $\widetilde{O}(d^2)$. A quick look at the chain and the rewinding strategy seems to suggest that such a d^2 complexity is

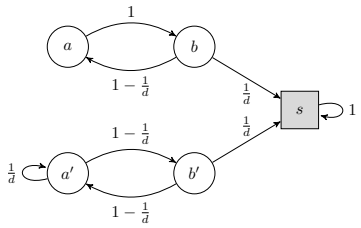


Figure 2.1 A five-state example.

essential; but we show later that a more sophisticated strategy actually distinguishes a from a' in $O(d)$ queries.

A simple algorithm with $\widetilde{O}(d^2)$ query complexity. We say a node in the tree is an A-node if its state is a or a' and a B-node if its state is b or b'. First, we claim that given a node v in the tree, we can determine with probability $1-\varepsilon$ if it is an A or B node with $O(d\log(1/\varepsilon))$ queries – we call this an A/B test. To do this, note that if v is an A-node, any child of v is non-sink (specifically, it is a B-node). Moreover, if v is a B-node, each child has a probability of $1/d$ of being a sink node. Therefore, opening $O(d\log(1/\varepsilon))$ children for a node v is sufficient for A/B testing v.

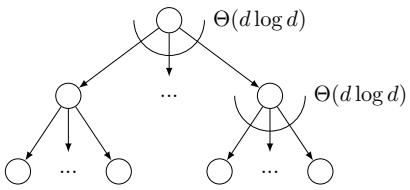


Figure 2.2 An $O(d^2\log^2 d)$-query strategy.

Next, having the A/B test available to us, we aim to assert whether a given A-node v has state a or a'. To do so, we first open $d\log(1/\varepsilon)$ children for v. If v has state a, then all of its children must be B-nodes. On the other hand, if v has state a', each child is an A-node with probability $1/d$, and so at least one must be an A-node with probability $1-(1-1/d)^{d\log(1/\varepsilon)} \geq 1-\varepsilon$. It then suffices to run the A/B test on these children to check if any of them is an A node. This takes total time $O(d^2\log^2(1/\varepsilon))$. Choosing $\varepsilon = 1/d^3$ so that all A/B tests are correct w.h.p., we get an overall query complexity of $O(d^2\log^2 d)$. The final queried tree is non-adaptive and is illustrated on the right.

Improving query complexity to $O(d)$. We now describe how we can improve over the algorithm above, and distinguish the a and a' states with just $O(d)$ queries. Here is the key insight. Suppose that we open a path of length $2d$ from the root and let us condition on not seeing the s node at all (which happens with constant probability). Then, if we start from the a state, we must alternatively visit a and b, finally arriving back at a since the length of the path is even. However, if we start from a', there is a constant probability that we go from a' to a' exactly once throughout the process since $P(a',a') = 1/d$ and the length of the walk is $2d$. If this event happens, we end up in state b'. Therefore, it suffices to run the A/B test on the last vertex of the path – if it happens to be B, we will be sure that we started from a'. Note that the success probability is a small constant, but we can boost it by

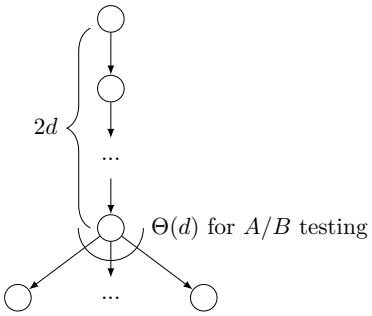


Figure 2.3 An $O(d)$-query strategy.

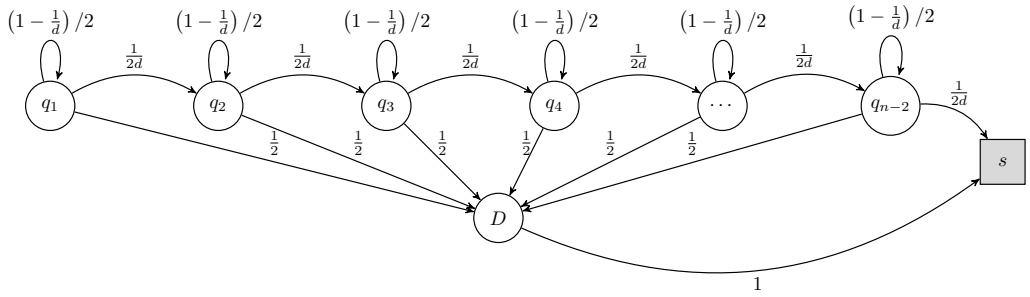


Figure 2.4 A canonical Markov chain for which there is a (large) polynomial gap between adaptive and non-adaptive strategies of distinguishing states q_1 and q_2.

repeating this process multiple times. The figure on the right shows the final query tree (for a single repetition of this process), which again is non-adaptive.

2.2 Example 2: The power of adaptivity

Our second example shows the power of adaptive rewinding strategies compared to non-adaptive ones. This example, illustrated in Figure 2.4, is later used to prove a polynomial gap between the non-adaptive and adaptive query complexities in the worst case (Theorem 1.2). We formalize this in Section 5, but here provide an intuition of how adaptivity helps.

Given a parameter $d \geq 1$, the Markov chain is formed by combining a "simple Markov chain" and a "dummy state". The simple Markov chain consists of a path of states, where each state transitions to the next with probability $\frac{1}{d}$, and stays in the same state otherwise. States can be easily distinguished in the simple Markov chain using $O(n^2 d)$ queries, where $n = |\Omega|$ is the number of states. To do so, it suffices to take random walks from the initial state and examine the time it takes to move to the sink (see Section 5 for the proof).

However, when combined with the dummy state, every state has a $\frac{1}{2}$ chance of jumping to the dummy state (otherwise, it proceeds as before). Then, the dummy state directly moves to the sink with probability 1. As all the states have the same probability of transitioning to the dummy state, moving to the sink through the dummy state reveals no information about the initial state.

Adaptive strategies can easily filter out the dummy state. That is, they can simulate a random walk on the simple Markov chain by checking if the walk has transitioned to the dummy state along the way, and rewinding back to the previous step if it has. As a result,

adaptive strategies retain the $O(n^2 d)$ query complexity even with the dummy state added.

The non-adaptive strategies cannot do the same, i.e. they cannot check whether a random walk has transitioned to the dummy state on the fly (they can only infer it after all the queries are made). Intuitively, to make up for this, they have to take many more random walks from the initial state, so as to guarantee that plenty of them reach the sink without moving through the dummy state. Therefore, the non-adaptive query complexity suffers when the dummy state is added and becomes (approximately) $\Omega(d^n)$. See Section 5 for the proof.

2.3 Technical Overview

Recall that our main goal is to bring a systematic understanding of distinguishing two candidate initial states in a known Markov chain. In this section, we provide a brief overview of our methods. There are two main aspects associated with our task: (1) computing rewinding strategies to distinguish between two initial states, and (2) proving lower bounds for the query complexity of this task. Together the two aspects allow us to establish the efficiency of our algorithm, i.e. that its run time is competitive with the optimal query complexity. Our approach to both aspects heavily utilizes partitions of the state set Ω.

Let $\mathcal{P}$ be a partition of the states, and for a state $a \in \Omega$ let $\mathcal{P}(a)$ denote the "class" of a in $\mathcal{P}$, i.e., the set within the partition $\mathcal{P}$ that contains a. As a subroutine, we study the problem of identifying the class of any given (hidden) state in $\mathcal{P}$, i.e., the task of computing $\mathcal{P}(X_0)$ for a given chain M and partition $\mathcal{P}$ (note that this is a generalization of the A/B test in Section 2.1). By choosing $\mathcal{P}$ carefully, this allows us to distinguish between two states $a, b \in \Omega$. Specifically, it suffices to identify the class of the initial state for some partition $\mathcal{P}$ that separates a and b, i.e., where $\mathcal{P}(a) \neq \mathcal{P}(b)$.

Intuitively, identifying the class within a given partition becomes more difficult as the partition becomes more refined. As an extreme case, the test for the trivial partition $\mathcal{P}_0 = \{\Omega \setminus \{s\}, \{s\}\}$ can be performed without any further queries in a canonical Markov chain since the partial observation at any state determines its class in $\mathcal{P}_0$. Our key tool in the design of algorithms is to assume that for some partition $\mathcal{P}_1$ we know how to identify the states, and then to use this algorithm to build a more refined partition $\mathcal{P}_2$ where we can identify states. Starting this with the trivial partition, we can use this method to work our way up to more and more refined partitions, till we reach a partition that separates a and b at which stage our problem would be solved. The exact details of this refinement step (and how we estimate their query complexity) are described below.

Recall that the goal of Theorem 1.1 is to develop a non-adaptive algorithm that distinguishes between two states a and b in time $O_n(1) \cdot \mathsf{QC}_M(a,b)^{O(n)}$, where n is the number of states and $O_n(1)$ suppresses terms dependent only on n. To do so, we define a weighted directed graph on the partitions: Namely, every partition $\mathcal{P}$ of Ω is a vertex of this graph and there is a weighted edge between every pair of partitions $\mathcal{P}_1 \to \mathcal{P}_2$ where $\mathcal{P}_2$ refines $\mathcal{P}_1$. The weights $w(\mathcal{P}_1, \mathcal{P}_2)$ are formally defined shortly. Intuitively, the main idea is to get a quantity that roughly allows us to determine the class of the initial state in $\mathcal{P}_2$ using approximately $w(\mathcal{P}_1, \mathcal{P}_2)$ calls to the class-identifier for $\mathcal{P}_1$. Note that if there is a path $\mathcal{P}_1 \to \mathcal{P}_2 \to \mathcal{P}_3$ in this graph, then this amounts to saying that classes of $\mathcal{P}_3$ can be identified using roughly $w(\mathcal{P}_1, \mathcal{P}_2) \cdot w(\mathcal{P}_2, \mathcal{P}_3)$ calls to the class identifier for $\mathcal{P}_1$. Thus, given this graph, we can easily devise a non-adaptive strategy for distinguishing between a and b.

For every partition $\mathcal{P}$ we compute the minimum distance from the trivial partition $\mathcal{P}_0$ (using this multiplicative measure of length of a path), and then use the partition $\mathcal{P}$ that separates a from b and has the smallest distance among all such $\mathcal{P}$. In particular, if we

let $\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_k = \mathcal{P}$ be the shortest path to $\mathcal{P}$, the test for each $\mathcal{P}_i$ can be recursively performed by calling the tests $\mathcal{P}_{i-1}$. This results in a test for $\mathcal{P}_k$ (which recall differentiates between a and b) that uses $c(\mathcal{P}_k) = \prod_{1 \le i \le k} w(\mathcal{P}_{i-1}, \mathcal{P}_i)$ tests. For Theorem 1.1, we also need to prove $c(\mathcal{P}_k) \le \mathsf{QC}_M(a,b)^{O(n)}$ which we discuss shortly, but first we describe how we compute $w(\mathcal{P}_1, \mathcal{P}_2)$.

To understand how a test for $\mathcal{P}_1$ can be used to identify the class in $\mathcal{P}_2$ of a given state, consider, as a warm-up, the original problem of distinguishing between a and b. Equipped with the test for $\mathcal{P}_1$, one can differentiate between a and b as follows: Given a state $x \in \{a, b\}$, we take many samples from the next state (rewinding back to x after each sample) and examine the distribution of their classes in $\mathcal{P}_1$. If the distribution of these classes for a and b are δ-apart in total variation distance, then x can be identified successfully with probability $1 - \varepsilon$ using $O(\log(1/\varepsilon)/\delta^2)$ samples. A similar approach can be taken to distinguish classes of $\mathcal{P}_2$ (instead of two states a and b). This is again captured by the total variation distance of certain distributions which gives us the weight $w(\mathcal{P}_1, \mathcal{P}_2)$ in the graph. Specifically, we can take

$$w(\mathcal{P}_1, \mathcal{P}_2) = \max_{\substack{c,d: \\ \mathcal{P}_1(c)=\mathcal{P}_1(d) \\ \mathcal{P}_2(c)\neq\mathcal{P}_2(d)}} \frac{1}{d_{\mathrm{TV}}^{\mathcal{P}}(c,d)^2},$$

where for $x \in \{c, d\}$, and $d_{\mathrm{TV}}^{\mathcal{P}}(c,d)$ denotes the total variation distance between next states from c and d when projected on $\mathcal{P}$. That is, letting X^c and X^d be the next state of the Markov chain conditioned on the last state being c and d respectively, $d_{\mathrm{TV}}^{\mathcal{P}}(c,d) := d_{\mathrm{TV}}(\mathcal{P}(X^c), \mathcal{P}(X^d))$. Thus, this expression captures the hardest to separate pair of states c and d from different classes of $\mathcal{P}_2$ (and not already separated in $\mathcal{P}_1$) after taking one step on the Markov chain starting at these states, using only a class identifier for $\mathcal{P}_1$.

Turning to our lower bound, recall that we wish to express the shortest path length in terms of the query complexity, i.e. to prove that $\prod_{1 \le i \le k} w(\mathcal{P}_{i-1}, \mathcal{P}_i)$ is at most

$$O_n(1) \cdot (\mathsf{QC}_M(a,b))^{O(|\Omega|)},$$

where $\mathcal{P}_0 \to \mathcal{P}_1 \cdots \to \mathcal{P}_k$ is the shortest path found by our algorithm. The argument has two main components, given a partition $\mathcal{P}$: (1) if two states c and d with $\mathcal{P}(c) = \mathcal{P}(d)$ are far in terms of $d_{\mathrm{TV}}^{\mathcal{P}}$, then $\mathcal{P}$ has a small outgoing edge separating c and d, and (2) if every two states within the same class of $\mathcal{P}$ are close in terms of $d_{\mathrm{TV}}^{\mathcal{P}}$, then it is difficult to distinguish any two states in the same class of $\mathcal{P}$. Note, in particular, that if such a $\mathcal{P}$ does not separate a and b, then it would serve as a "certificate" for the difficulty of distinguishing between them.

We begin by explaining how these components can be applied to obtain upper bounds on the shortest path (equivalently, lower bounds on the query complexity). Consider any partition $\mathcal{P}$, where a and b are not separated. Given that a and b are distinguishable with $\mathsf{QC}_M(a,b)$ queries, we can employ (2) to show there are two states c and d with $\mathcal{P}(c) = \mathcal{P}(d)$ such that $d_{\mathrm{TV}}^{\mathcal{P}}(c,d) = \Omega(1/\mathsf{QC}_M(a,b))$. Then, using (1), we can infer that $\mathcal{P}$ has an outgoing edge of weight at most $O_n(1)/d_{\mathrm{TV}}^{\mathcal{P}}(c,d)^2 = O_n(1) \cdot \mathsf{QC}_M(a,b)^2$. As a result, we can start at $\mathcal{P}_0$ and iteratively take the smallest outgoing edge, which has weight at most $O_n(1) \cdot \mathsf{QC}_M(a,b)^2$, until we reach a partition that separates a and b. The path can take a length of at most $n-1$ since each partition is refined by the next. This gives us a path of length at most $O_n(1) \cdot \mathsf{QC}_M(a,b)^{2n}$ from $\mathcal{P}_0$ to a partition that separates a and b.

Now we discuss each component separately. First, consider a partition $\mathcal{P}$ with two states c and d in the same class of $\mathcal{P}$ that have $d_{\mathrm{TV}}^{\mathcal{P}}(c,d) = D$. Let G be an auxiliary graph where

the vertices are the Markov chain states, and two states are connected if they are closer than D/n in $d_{\mathrm{TV}}^{\mathcal{P}}$. We construct a refinement $\mathcal{P}'$ of $\mathcal{P}$ by letting two states be in the same class of $\mathcal{P}'$ if they are in the same class of $\mathcal{P}$ and the same connected component of G. Observe that $\mathcal{P}'$ separates c and d since $d_{\mathrm{TV}}^{\mathcal{P}}$ forms a metric, and c and d are more are D apart, whereas any two directly connected states are at most D/n apart. Furthermore, the only states that are newly separated by $\mathcal{P}$ are at least D/n apart. Therefore, $\mathcal{P}$ has an outgoing edge of weight at most $(n/D)^2$ to $\mathcal{P}'$.

For the second component, take a partition $\mathcal{P}$ where any two states in the same class of $\mathcal{P}$ are at most θ apart in $d_{\mathrm{TV}}^{\mathcal{P}}$. We show that for any two states a and b with $\mathcal{P}(a) = \mathcal{P}(b)$, it holds $\mathsf{QC}_M(a, b) = \Omega(1/\theta)$. Given two possible initial states a and b to distinguish, we consider two instances of running the rewinding strategy: one with a, and one with b. Let T_a and T_b be the query trees that the rewinding strategy creates in each case. We couple T_a and T_b such that they look the same to the strategy with a large probability. Here, by looking the same, we mean that the states generated in T_a and T_b yield the same observations, i.e. they are both sink or non-sink. In addition, the coupling is such that T_a and T_b are isomorphic w.r.t. $\mathcal{P}$, meaning the corresponding nodes in T_a and T_b have states that are in the same class of $\mathcal{P}$, with a large probability.

We construct our couplings in an inductive manner. In every step, a state $u_a \in T_a$ and the corresponding state $u_b \in T_b$ are chosen by the rewinding strategy to derive the next step. At this point, we need a method to couple the next step drawn from u_a and u_b such that they are both sink or non-sink with a large probability. Such a coupling is not possible for arbitrary states u_a and u_b (e.g. $\mathcal{P}_0(u_a) = \mathcal{P}_0(u_b)$ does not imply the same is true for their next step with a large probability). To do so, we employ the additional property that $\mathcal{P}(a) = \mathcal{P}(b)$, and show that the drawn states can be coupled such that they are in the same class of $\mathcal{P}$ with probability $1 - \theta$. Intuitively, if a rewinding strategy distinguishes between a and b, it should make at least $\Omega(1/\theta)$, so that the above scheme fails and the drawn states are in different classes of $\mathcal{P}$.

We remark that something as strong as the shortest path algorithm is not required for Theorem 1.1. While it is intuitive to use the shortest path as it corresponds to "the cheapest way" for testing $\mathcal{P}$ in our graph, something as simple as Prim's algorithm also works. That is, to prove Theorem 1.1, we essentially show that there exists a partition $\mathcal{P}$ separating a and b which is reachable from $\mathcal{P}_0$ through edges of weight at most $O_n(1) \cdot \mathsf{QC}_M(a, b)^2$. Therefore, Prim's algorithm can find such a $\mathcal{P}$, by starting from $\mathcal{P}_0$ and in each step, adding the minimum outgoing edge to the tree. Also, note that while the shortest path algorithm may use fewer queries in many cases, in the worst case, it uses (asymptotically) the same number of queries as Prim's algorithm (Theorem 1.2).

3 The Formal Model

We use $\mathbb{Z}^{\geq 0}$ to denote the set of non-negative integers. We use $d_{\mathrm{TV}}(\cdot, \cdot)$ to denote the total variation distance between two distributions.

We start by reviewing the standard notions of (partially observable) Markov chains. A sequence of random variable $X_0, X_1, X_2, \ldots$ with $X_t \in \Omega$ for all t is a *Markov chain* if

there exists a transition matrix $P : \Omega \times \Omega \to [0, 1]$ such that for every t and all $x, y \in \Omega$, $\Pr(X_{t+1} = y \mid X_t = x) = P(x, y)$; we write $P(x, \cdot)$ for the distribution from x to the next state. [2] A sequence $Z_0, Z_1, \ldots$ with $Z_t \in \Sigma$ is a *partially observable (p.o.) Markov chain* if

[2] We note that such Markov chains are called *time-invariant* since transition probabilities, conditioned on the current state, are independent of time.

there exists a Markov chain $X_0, X_1, \ldots$ and a function $O : \Omega \to \Sigma$ such that $Z_t = O(X_t)$ for all t. (While a general theory would allow probabilistic functions O, we restrict our attention to deterministic functions in this work.) We refer to Ω as the state space and Σ as the observable space. Note that a partially observable chain M is given by a triple (Ω, P, O).

Next, we turn to the central objects of study in this paper, namely Markov chains with rewinding. Before doing so, we briefly touch upon two views of a rewinding strategy. The formal view is simply as a function that maps a history of observations to a non-negative number (which we view as the number of steps we rewind the current time by). An alternate view is that the rewinding strategy builds a tree where, at each time step, the algorithm picks a node of the current tree, and the next time step produces a new child for this node whose state is independent of that of all other nodes, conditioned on the parent's state. The following definition uses the formal view, but we often switch to the tree view in our analyses.

▶ **Definition 3.1** (Markov Chains with Rewinding). *For a partially observable Markov chain $M = (\Omega, P, O)$ with observable space Σ, a rewinding strategy is a function $A : \Sigma^* \to \mathbb{Z}^{\geq 0}$. A Markov chain $M = (\Omega, P, O)$ in initial state X_0 with rewinding strategy A generates a sequence of random variables $Z_0, Z_1, Z_2, \ldots$, with companion variables $X_1, X_2, \ldots$ as follows: For every $t \geq 0$, we let $Z_t = O(X_t)$ and $X_{t+1} \sim P(X_{t'}, \cdot)$ where $t' = \max\{0, t - A(Z_0, \ldots, Z_t)\}$. We say that a sequence of random variables $Z_0, Z_1, \ldots,$ is a Markov chain with rewinding if there exists a chain M, an initial state X_0, and a rewinding strategy A that generates this sequence.*

Markov chains with rewinding strictly generalize the class of Markov chains, which correspond to *passive observation*, i.e., the rewinding strategy $A(\cdot)$ is the constant function 0, or equivalently, the rewinding tree is a path (and hence "chain"). Another class of rewinding strategies, widely used in algorithm design, picks some threshold $\tau \in \mathbb{Z}^{\geq 0}$ and sets $A(Z_0, \ldots, Z_t) = t$ if $t = 0 \pmod{\tau}$ and $A(Z_0, \ldots, Z_t) = 0$ otherwise. Equivalently, the rewinding tree consists of paths of length τ intersecting at the root. Such strategies may be termed *resetting strategies*. A simple adaptation of resetting is inspired by the "go-with-the-winners" algorithm [1] is the following: $A(Z_0, \ldots, Z_t) = 0$ if $t \neq 0 \pmod{\tau}$ and $\arg\max_\ell \{Z_{t-\ell}\}$ otherwise. That is, the chain-rewinder seeks to maximize the observed variable, and once in every τ steps, it rewinds to a previous state with the highest observable.

An important subclass of strategies which we study in this work are *non-adaptive* rewinding strategies, where $A(Z_0, \ldots, Z_t) = A'(t)$, i.e., the rewinding strategy is a fixed function of the length and does not depend on the actual observations. Note that resettable strategies are special cases of non-adaptive strategies.

The main property of Markov chains we explore is the *identifiability* of the initial state. Specifically, given a p.o. Markov chain M and two candidate initial states $a, b \in \Omega$, we ask how long a rewinding strategy has to run (or how many "queries" it must make to M) to distinguish between these two starting states. The following definition formalizes this notion. It only considers identifiability from an information-theoretical point of view, i.e. the query complexity:

▶ **Definition 3.2** (Query Complexity). *Given a p.o. Markov chain M and rewinding strategy A, let $Z_0^a, Z_1^a, Z_2^a, \ldots$ denote the Markov chain with rewinding obtained from the initial state $X_0 = a$ and let $Z_0^b, Z_1^b, Z_2^b, \ldots$ correspond to the case $X_0 = b$. We say that (M, A, a, b) are (ε, T)-identifiable if $d_{\mathrm{TV}}((Z_0^a, \ldots, Z_T^a), (Z_0^b, \ldots, Z_T^b)) \geq \varepsilon$. The* query complexity *of distinguishing a from b in chain M with rewinding strategy A, denoted $\mathsf{QC}_M^A(a, b)$, is the minimum T such that (M, A, a, b) are $(1/3, T)$-identifiable. Having this, we define:*

- *the (adaptive) query complexity of distinguishing a from b to be:*

$$\mathsf{QC}_M(a,b) = \min_A \mathsf{QC}_M^A(a,b),$$

- *and define the analogous notion for non-adaptive rewinding strategies. Namely,*

$$\mathsf{NAQC}_M(a,b) = \min_{\text{non-adaptive } A} \mathsf{QC}_M^A(a,b).$$

We also consider identifiability from a computational standpoint. That is, we study the time complexity for algorithms that identify the initial state. Such an algorithm produces the rewinding strategy and queries the Markov chain accordingly. Then, it identifies the initial state based on all the observations. We call such algorithms *state identification algorithms*.

▶ **Definition 3.3** (State Identification Algorithms). *The input of a state identification algorithm consists of a p.o. Markov chain $M = (\Omega, P, O)$, two candidate initial states $a, b \in \Omega$, and a hidden initial state $X_0 \in \{a, b\}$. An (adaptive) state identification algorithm in every step $t \geq 0$, computes the rewinding strategy $A(Z_0, \ldots, Z_t)$, after which X_{t+1} is sampled according to $P_{X|Y=X_{t'}}$ with $t' = \max\{0, t - A(Z_0, \ldots, Z_t)\}$, and the observation $Z_{t+1} = O(X_{t+1})$ is revealed to the algorithm. A non-adaptive algorithm computes the rewinding strategy before sampling any states. The algorithm decides the number of steps T, and eventually outputs the identified initial state (a or b). We say the algorithm successfully identifies the initial state if the output is correct with probability $2/3$.*

Finally, we define a special subclass that we refer to as *canonical* p.o. Markov chains. While general p.o. Markov chains are of interest, we state our results in terms of canonical p.o. Markov chains for simplicity. We show later, in Theorem 6.2, that canonical chains capture all chains in our context.

▶ **Definition 3.4** (Canonical p.o. Markov Chains). *We say that p.o. Markov chain $M = (\Omega, P, O)$ is a* canonical *p.o. Markov chain if there exists a state $s \in \Omega$ that is a sink state (i.e., $P(s,s) = 1$ and equivalently $P(s,y) = 0$ for all $y \neq s$), and $O(a) = 1$ if $a = s$ and 0 otherwise.*

Thus, in a canonical p.o. Markov chain, the only observable signal is whether the chain has reached the sink or not; and once one reaches the sink, no further observations reveal information. Such chains are the hardest to learn and hence our focus.

3.1 Partitions

In the rest of this section, we introduce the notion of state partitions, i.e. partitions of Ω, a key tool both for designing rewinding strategies to distinguish between states, and proving lower bounds on the query complexity of this task. Throughout the section we use n as a shorthand for $|\Omega|$, $p(a,b)$ for $P(a,b) = \Pr(X_{t+1} = b \mid X_t = a)$, and $p(a,C)$ for $\sum_{b \in C} p(a,b)$. We use s to denote the sink state.

▶ **Definition 3.5** (Partition). *Given a set Ω, a partition $\mathcal{P}$ is a collection of sets such that each element $a \in \Omega$ appears in exactly one set of the partition. We often refer to these sets as classes, and use $\mathcal{P}(a)$ to denote the class of a in $\mathcal{P}$.*

▶ **Definition 3.6** (Total Variation Distance w.r.t. Partitions). *For two states a and b, and a partition $\mathcal{P}$, the total variation distance of $P(a,\cdot)$ and $P(b,\cdot)$ w.r.t. to $\mathcal{P}$ is*

$$d_{TV}^{\mathcal{P}}(a,b) := \frac{1}{2} \sum_{C \in \mathcal{P}} |p(a,C) - p(b,C)| = \max_{A \subseteq \mathcal{P}} \sum_{C \in A} p(b,C) - p(a,C).$$

Here, A is a collection of the classes in $\mathcal{P}$.

▶ **Lemma 3.7.** *Let $\mathcal{P}$ be a partition of Ω. Given two states a and b, and oracle access to $\mathcal{P}(x)$ for any state $x \in \Omega$, there exists a non-adaptive rewinding strategy that successfully distinguishes between initial states a and b with probability $1 - \varepsilon$ using*

$$O\left(\frac{\log 1/\varepsilon}{\left(d_{\mathrm{TV}}^{\mathcal{P}}(a,b)\right)^2}\right)$$

queries.

Proof. Let $d = d_{\mathrm{TV}}^{\mathcal{P}}(a,b)$, and let $A \subseteq \mathcal{P}$ be the collection of classes where the difference in the distributions is maximized:

$$A := \arg\max_{A \subseteq \mathcal{P}} \sum_{C \in A} p(b,C) - p(a,C).$$

Note that by definition $p(b,A) - p(a,A) = d$. Given a state $x \in \{a,b\}$, we draw $\frac{2\log 1/\varepsilon}{d^2}$ samples of the next step of x (according to distribution $P(x,\cdot)$), and let ρ be the fraction of them that land in A. If the state is a, the expected value of ρ is $p(a,A)$; if the state is b, the expected value is $p(b,A)$; and the difference between the two is equal to d. We declare that the state is a if $\rho < p(a,A) + \frac{d}{2}$ and that it is b otherwise. The probability of error when $x = a$ is equal to:

$$\Pr\left(\rho \geq p(a,A) + \frac{d}{2}\right) \leq \exp\left(-2 \cdot \frac{2\log 1/\varepsilon}{d^2} \cdot \left(\frac{d}{2}\right)^2\right) \leq \varepsilon.$$

Similarly, the probability of error when $x = b$ is equal to:

$$\Pr\left(\rho < p(a,A) + \frac{d}{2}\right) = \Pr\left(\rho < p(b,A) - \frac{d}{2}\right) \leq \varepsilon.$$ ◀

4 A Polynomially Optimal Non-Adaptive Algorithm

4.1 A Non-Adaptive Algorithm with Polynomial Queries

In this section, we present a non-adaptive algorithm that distinguishes any two (distinguishable) states a and b of an n-state canonical Markov chain $M = (\Omega, P)$ (see Definition 3.4) using $O_n(1) \cdot \mathsf{QC}_M(a,b)^{O(n)}$ queries, where recall $\mathsf{QC}_M(a,b)$ is the optimal query complexity of distinguishing a from b with a possibly adaptive rewinding strategy, and $O_n(1)$ is used to suppress terms dependent only on n. In particular, this implies that the gap between adaptive and non-adaptive query strategies is only polynomial so long as the number of states of the Markov chain is constant. We later show in Section 5 that this is the best one can hope for: there are choices of M, a, b where any non-adaptive rewinding strategy requires $\mathsf{QC}_M(a,b)^{\Omega(n)}$ queries.

Formally, we show:

▶ **Theorem 4.1.** *Let $M = (\Omega, P)$ be a canonical p.o. Markov chain with $n = |\Omega|$ states. There is a non-adaptive algorithm (Algorithm 1) that distinguishes any distinguishable states $a, b \in \Omega$ in time*

$$O_n(1) \cdot (\mathsf{QC}_M(a,b) \log \mathsf{QC}_M(a,b))^{2(n-2)} = O_n(1) \cdot \mathsf{QC}_M(a,b)^{O(n)}.$$

This is equivalent to Theorem 1.1.

4.2 An Informal Overview of the Algorithm

The algorithm defines a weighted directed graph $G_M = (V, E)$ where each vertex of the graph corresponds to a partitioning of Ω, the state space of the Markov chain. There is an edge from $\mathcal{P} \in V$ to $\mathcal{P}' \in V$ iff $\mathcal{P}'$ is a refinement of $\mathcal{P}$. Informally speaking, it would be helpful to view each partitioning $\mathcal{P}$ as a *test* that reveals which class of $\mathcal{P}$ a hidden state belongs to. With this view, we define the weight $w(\mathcal{P}, \mathcal{P}')$ of the directed edge $(\mathcal{P}, \mathcal{P}') \in E$ such that it upper bounds the cost of solving test $\mathcal{P}'$ having free access to test $\mathcal{P}$. Once we define this graph G_M, we compute the shortest path tree on G_M (e.g. using Dijkstra's algorithm) starting from the trivial partitioning $\mathcal{P}_0 = \{\Omega \setminus \{s\}, \{s\}\}$ that only separates the sink from the other states. Let $\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_k$ be the unique path in the tree from $\mathcal{P}_0$ to a partitioning $\mathcal{P}_k$. We will define the cost of $\mathcal{P}_k$ as

$$c(\mathcal{P}_k) = \prod_{i=0}^{k-1} w(\mathcal{P}_i, \mathcal{P}_{i+1}).$$

We prove that each test $\mathcal{P}$ can indeed be solved with (slightly more than) $c(\mathcal{P})$ queries. To distinguish state a from b, the algorithm takes the partitioning $\mathcal{P}$ that separates a from b and has the minimum cost $c(\mathcal{P})$. Our analysis shows that $c(\mathcal{P})$ can be upper bounded by $O_n(1) \cdot \mathsf{QC}_M(a, b)^{O(n)}$.

4.3 The Formal Argument

▶ **Definition 4.2** (Source Partition). *The **source partition** $\mathcal{P}_0$ is the trivial partition where the sink is a singleton class and all the remaining states form another class, i.e.* $\mathcal{P}_0 = \{\Omega \setminus \{s\}, \{s\}\}$.

For the purposes of this section, we only consider the partitions that separate s from every other state, i.e. partitions that refine $\mathcal{P}_0$.

▶ **Definition 4.3** (Partition Graph). *Given a canonical Markov chain $M = (\Omega, P)$, let the partition graph G_M be a weighted directed graph where each node is a partition of the states Ω. For a partition $\mathcal{P}_1$ and a refinement $\mathcal{P}_2$, there is a directed edge from $\mathcal{P}_1$ to $\mathcal{P}_2$ with weight*

$$w(\mathcal{P}_1, \mathcal{P}_2) = \max_{\substack{a,b: \\ \mathcal{P}_1(a)=\mathcal{P}_1(b) \\ \mathcal{P}_2(a)\neq\mathcal{P}_2(b)}} 1/\left(d_{\mathrm{TV}}^{\mathcal{P}_1}(a, b)\right)^2.$$

The lengths of the paths are defined multiplicatively. That is, the length of a path is the product of the weights of its edges. For a partition $\mathcal{P}$, its cost $c(\mathcal{P})$ is equal to the length of the shortest path from $\mathcal{P}_0$ to $\mathcal{P}$.

Below is a formal overview of the algorithm. The details of using the shortest paths to distinguish between states are given in Lemma 4.8. We use the notion of trees for rewinding. That is, if X_t is sampled by rewinding to $X_{t'}$ (i.e. according to distribution $P_{X|Y=X_{t'}}$), then $X_{t'}$ is the parent of X_t in the tree.

To analyze the query complexity of Algorithm 1, we show there exists a path of length $O_n(1) \cdot \mathsf{QC}(a, b)^{2(n-2)}$ to a partition that separates a and b (Lemma 4.7). Then, we show how this path can be used to distinguish between a and b (Lemma 4.8).

First, we prove some auxiliary claims. The following lemma essentially upper-bounds the weight of the smallest outgoing edge of a partition $\mathcal{P}$, based on two states a and b with $\mathcal{P}(a) = \mathcal{P}(b)$.

Algorithm 1 A non-adaptive algorithm with polynomial queries.

Input: A Markov chain $M = (\Omega, P)$, and two states a and b
Let G_M be the partition graph of the Markov chain as in Definition 4.3.
Compute the (multiplicative) shortest path tree on G_M rooted at the trivial partition $\mathcal{P}_0$.
Let $\mathcal{P}$ be a partition separating a and b that minimizes $c(\mathcal{P})$.
Let $\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_k = \mathcal{P}$ be the shortest path to $\mathcal{P}$.
Let $\varepsilon = \Theta_n\left(\frac{1}{c(\mathcal{P}) \log c(\mathcal{P})}\right)$.
Query a tree of height k, where the vertices at height i have degree $\Theta_n(\log(1/\varepsilon) \cdot w(\mathcal{P}_{i-1}, \mathcal{P}_i))$.
Use the observations in the tree to identify the class of each vertex at height i in $\mathcal{P}_i$ recursively (see Lemma 4.8 for the details).

▶ **Lemma 4.4.** *Take a partition $\mathcal{P}$ and any two states $a, b \in \Omega$ such that $\mathcal{P}(a) = \mathcal{P}(b)$. Then, there exists another partition $\mathcal{P}'$ that refines $\mathcal{P}$ and*

$$w(\mathcal{P}, \mathcal{P}') \leq \left(\frac{n-1}{d_{\mathrm{TV}}^{\mathcal{P}}(a,b)}\right)^2.$$

▶ **Corollary 4.5.** *For a partition $\mathcal{P}$, if $\mathcal{P}$ has no outgoing edge of weight at most h in the partition graph, then for any two states a and b in the same class of $\mathcal{P}$, it holds that*

$$d_{\mathrm{TV}}^{\mathcal{P}}(a,b) < \frac{n-1}{\sqrt{h}}.$$

Proof. Follows directly from Lemma 4.4. ◀

The following claim states that if there exists a partition $\mathcal{P}$ such that every two states in the same class are closer than θ w.r.t. $d_{\mathrm{TV}}^{\mathcal{P}}$, then distinguishing any two states in the same class of $\mathcal{P}$ is difficult (roughly speaking, requires $1/\theta$ queries).

▷ Claim 4.6. For a partition $\mathcal{P}$ that does not separate a and b, let

$$\theta(\mathcal{P}) \coloneqq \max_{\substack{x,y: \\ \mathcal{P}(x)=\mathcal{P}(y)}} d_{\mathrm{TV}}^{\mathcal{P}}(x,y).$$

Then, any (possibly adaptive) rewinding strategy that uses Q queries, can successfully distinguish between a and b with probability at most $\frac{1}{2} + \frac{Q \cdot \theta(\mathcal{P})}{2}$.

Combining Corollary 4.5 and Claim 4.6, for a partition $\mathcal{P}$ with $\mathcal{P}(a) = \mathcal{P}(b)$, we can upper-bound the smallest outgoing edge of $\mathcal{P}$, and prove there exists a short path from $\mathcal{P}_0$ to a partition that separates a and b.

▶ **Lemma 4.7.** *There exists a path of length $O_n(1) \cdot \mathsf{QC}(a,b)^{2(n-2)}$ from $\mathcal{P}_0$ to a partition that separates a and b.*

The following lemma describes how a path of length Q can be used to distinguish the states with (slightly more than) Q queries.

▶ **Lemma 4.8.** *Given a path $\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_k$ of length Q, one can identify the class of the initial state in $\mathcal{P}_k$ with $O_n(1) \cdot Q \log^k Q$ queries,[3] using a non-adaptive tree of height k where*

[3] recall $k \leq n-2$

on the i-th level the degree of every node is chosen proportionally to $w(\mathcal{P}_{i-1}, \mathcal{P}_i)$ and the class of each node on that level is determined w.r.t. $\mathcal{P}_i$.

Finally, we put Lemmas 4.7 and 4.8 together to derive Theorem 4.1.

Proof of Theorem 4.1. By Lemma 4.7, for any two states a and b, there exists a path of length at most $\mathsf{QC}(a,b)^{2(n-2)}$ ending at a partition that separates a and b. Therefore, the partition $\mathcal{P}$ obtained by computing shortest paths in step 4 of the algorithm has cost at most $\mathsf{QC}(a,b)^{2(n-2)}$. Lemma 4.8 shows how the shortest path to $\mathcal{P}$ can be used in steps 7–8, to identify the class of the initial state in $\mathcal{P}$ using $O_n(1) \cdot (\mathsf{QC}(a,b) \log \mathsf{QC}(a,b))^{2(n-2)}$ queries, hence distinguishing between a and b.

Regarding the runtime, observe that the strategy is implicitly (non-adaptively) computed in $O_n(1)$. The number of state partitions (number of vertices in G_M) is at most $n^n = O_n(1)$. For an edge $\mathcal{P} \to \mathcal{P}'$, the weight can be computed in time $O(n^3)$ by iterating over every relevant pair of states and computing the distance $d_{\mathrm{TV}}^{\mathcal{P}}$ in $O(n)$. Then the shortest path tree can be computed using a polynomial-time algorithm (e.g. Dijkstra's). After the shortest path is computed, the degree on each level of the query tree is determined. For each node of height i, its calls in $\mathcal{P}_i$ can be determined in time $O(2n^2 \cdot \log(1/\varepsilon) \cdot w(\mathcal{P}_{k-1}, \mathcal{P}_k))$, i.e. proportional to the degree. Therefore, the run time of the algorithm is $O_n(1) \cdot (\mathsf{QC}(a,b) \log \mathsf{QC}(a,b))^{2(n-2)} = O_n(1) \cdot (\mathsf{QC}(a,b))^{O(n)}$. ◀

5 The Polynomial Gap of Adaptivity

In this section, we prove the following theorem which directly implies Theorem 1.2.

▶ **Theorem 5.1.** *For any parameters $n, d \geq 1$, there exists a canonical Markov chain $M = (\Omega, P)$ with $n = |\Omega|$ states and two states $a, b \in \Omega$ such that*

$$\mathsf{QC}_M(a,b) = O(n^2 d), \qquad \mathsf{NAQC}_M(a,b) = \Omega(d^{(1-o(1))n}).$$

We remark that the optimal rewinding strategy, which distinguishes between a and b using $O(n^2 d)$ queries, can be trivially turned into an algorithm with the same run time. Therefore, Theorem 5.1 implies the same gap between the optimal adaptive and non-adaptive run times of state identification algorithms. As such, in this section, we abuse the notation and use the terms rewinding strategy and algorithm interchangeably.

We prove the claim given $d > 0$ and, the Markov chain M appeared in Figure 2.4. In this Markov chain, the states form a directed path toward the sink state, with states labeled as $q_1, ..., q_{n-2}$ from left to right. We label the last state as D. For all $i \in [n-3]$, state q_i goes to q_{i+1} with probability $\frac{1}{2d}$. The state q_{n-2} goes to the sink state s with the same probability. Each state in $q_1, ..., q_{n-2}$ goes to itself with probability $\left(1 - \frac{1}{d}\right)/2$. All $q_1, ..., q_{n-2}$ go to the state D with probability $\frac{1}{2}$, while state D goes to the sink with probability 1.

We prove that there exists an adaptive algorithm distinguishing q_1 and q_2 with $O(n^2 d)$ queries, while any non-adaptive algorithm needs at least $\Omega(d^{(1-o(1))n})$ queries. The intuition is as follows. An adaptive algorithm can easily test whether each new state opened is the D state or not by simply doing a test with constant queries, and opening another child instead to continue its path. This is not possible for non-adaptive algorithms that come across the D state with probability $\frac{1}{2}$, and when this happens the algorithm intuitively loses any information on the state it started from. So it needs a lot of paths to account for this loss of information.

▶ **Lemma 5.2.** *There exists an adaptive algorithm that distinguishes q_1 and q_2 in Markov chain M of Figure 2.4 with $O(n^2 d)$ queries.*

▶ **Lemma 5.3.** *Any non-adaptive algorithm distinguishing q_1 and q_2 in Markov chain M of Figure 2.4 needs $\Omega(d^{(1-o(1))n})$ queries.*

Proof of Theorem 5.1. Follows from combining Lemmas 5.2 and 5.3. ◀

6 Generality of Canonical Markov Chains

In this section, we show that identifying the initial state of a canonical Markov chain (Definition 3.4) is (essentially) as hard as the same task in any partially observable Markov chain via a reduction. That takes into account both the query complexity of the rewinding strategy and the runtime of the algorithm. For a state identification algorithm $\mathcal{A}$ we write $\mathrm{T}(\mathcal{A})$ for the time it spends outside the oracle that samples the next state.

In the generalized version, a Markov chain $M = (\Omega, P)$ is complemented by an observation function $O : \Omega \to \Sigma$, i.e. every state $x \in \Omega$ in the Markov chain outputs an observation $O(x) \in \Sigma$. The states remain hidden, but in every step, the algorithm can see the observation and use it to identify the initial states. A canonical Markov chain can be expressed in this generalized version by letting the set of possible observations be $\Sigma = \{0, 1\}$, and letting $O(x) = \mathbf{1}(x = s)$ where s is the sink state. That is, the observation made at any state is whether it is the sink state. First, We show that any adaptive strategy taking Q queries in the generalized model can be emulated in the canonical Markov chain using $O(|\Sigma|Q)$ queries. Secondly, we show that for non-adaptive algorithms, the gap between the query complexity of canonical and generalized Markov chains is at most near-quadratic.

Formally, we define a reduction from a generalized Markov chain M to a canonical Markov chain $\hat{M}$ below. See Figures 6.1 and 6.2 for an example of this reduction.

▶ **Definition 6.1** (Reduction to Canonical Markov chains)**.** *Let $M = (\Omega, P)$ be a Markov chain with state space Ω, transition matrix P, and an observation function $O : \Omega \to \Sigma$ that maps each state to an observable output. Let the emulated Markov chain of M be a canonical Markov chain $\hat{M} = (\hat{\Omega}, \hat{P})$ with state space $\hat{\Omega}$ and transition matrix $\hat{P}$ as follows:*

For each state $x \in \Omega$, there exists a unique corresponding state $\hat{x} = \phi(x)$ in $\hat{\Omega}$, where $\phi : \Omega \to \hat{\Omega}$ is an injective mapping. Let $k = |\Sigma|$ and and $\Sigma = \{\sigma_1, \ldots, \sigma_k\}$. Given a parameter $0 < q < 1$ we have

1. *For every pair of states $x, x' \in \Omega$ with transition probability $p_{x,x'}$ in P, we introduce an extended path in $\hat{M}$ consisting of a series of intermediate states $d_1^{x,x'}, d_2^{x,x'}, \ldots, d_{k-1}^{x,x'}$ with transition probability 1 between consecutive states, such that $\hat{x}$ transitions to $\hat{x}'$ via these intermediate states. $\hat{x}$ transitions to $d_1^{x,x'}$ with probability $qp_{x,x'}$ and $d_{k-1}^{x,x'}$ transitions to $\hat{x}'$ with probability 1.*
2. *To emulate the observation function O, we introduce a set of **special states** $\sigma_1, \sigma_2, \ldots, \sigma_k = s$ in $\hat{M}$, each representing a unique observation in Σ. s is the sink state of the canonical Markov chain $\hat{M}$. Each state $\hat{x} \in \hat{\Omega}$ in $\hat{M}$ transitions to a special state corresponding to its observation in M with probability $1 - q$. Special state σ_i transitions to σ_{i+1} with probability 1 for $i \in [k - 1]$.*

▶ **Theorem 6.2.** *Let $M = (\Omega, P)$ be a Markov chain in the generalized model with an observation function $O : \Omega \to \Sigma$, Then, there exists a canonical Markov chain $\hat{M} = (\hat{\Omega}, \hat{P})$ states and an injective mapping of the states $\phi : \Omega \to \hat{\Omega}$, such that for any two states $a, a' \in \Omega$,*

$$\mathsf{QC}_{\hat{M}}(\phi(a), \phi(a')) = \Theta\big(|\Sigma|\mathsf{QC}_M(a, a')\big).$$

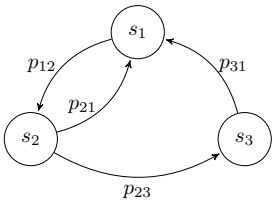


Figure 6.1 Represents a general Markov chain M with observations $\Sigma = \{\sigma_1, \sigma_2, \sigma_3\}$, where $O_{s_1} = \sigma_1, O_{s_2} = \sigma_2$, and $O_{s_3} = \sigma_2$.

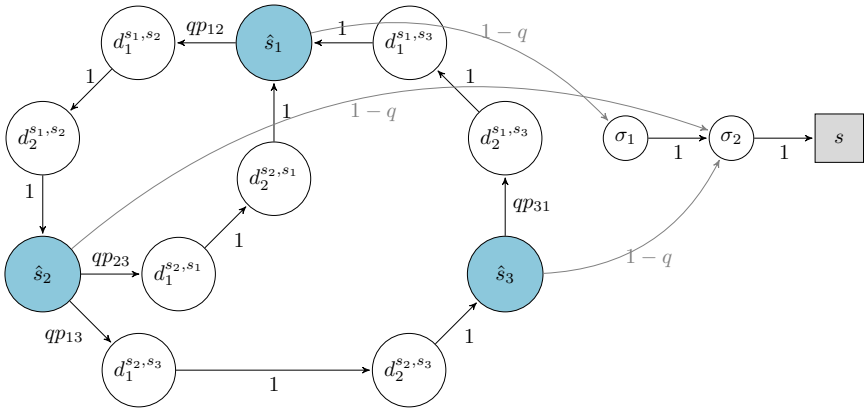


Figure 6.2 Represents a canonical Markov chain $\hat{M}$, with parameter $0 < q < 1$, where there is an injection ϕ that reduces distinguishing states a and a' in M to distinguishing $\phi(a)$ and $\phi(a')$ in $\hat{M}$.

▶ Remark 6.3. This reduction preserves running time up to the same multiplicative overhead: for any adaptive state identification algorithm $\mathcal{A}$ for M that makes Q queries and runs in time $Time(\mathcal{A})$, there is an algorithm $\hat{\mathcal{A}}$ for $\hat{M}$ that runs in time $Time(\mathcal{A}) + O(kQ)$, where $k = |\Sigma|$. Also, any adaptive algorithm $\hat{\mathcal{A}}$ for $\hat{M}$ with Q' queries and time $Time(\hat{\mathcal{A}})$ yields an algorithm for M with $\Omega(Q'/k)$ queries and time $Time(\hat{\mathcal{A}}) + O(Q')$.

▶ **Theorem 6.4.** *Let $M = (\Omega, P)$ be a Markov chain in the generalized model with an observation function $O : \Omega \to \Sigma$, Then, there exists a canonical Markov chain $\hat{M} = (\hat{\Omega}, \hat{P})$ states and an injective mapping of the states $\phi : \Omega \to \hat{\Omega}$, such that for any two states $a, a' \in \Omega$,*

$$\Omega\big(|\Sigma|\mathsf{NAQC}_M(a,a')\big) \leq \mathsf{NAQC}_{\hat{M}}(\phi(a),\phi(a')) \leq \tilde{O}\big(|\Sigma|\mathsf{NAQC}_M(a,a')^2\big).$$

▶ Remark 6.5. If a non-adaptive algorithm $\mathcal{A}$ for Markov chain M uses Q queries and runs in time $Time$, then the induced non-adaptive algorithm $\widehat{\mathcal{A}}$ for $\hat{M}$ can be implemented with $\tilde{O}(kQ^2)$ queries and total running time $Time(\mathcal{A}) + \tilde{O}(kQ^2)$, where $k = |\Sigma|$. Also, any non-adaptive $\hat{\mathcal{A}}$ for $\hat{M}$ with Q' queries and time $Time(\hat{\mathcal{A}})$ yields a non-adaptive algorithm for M with $\Omega(Q'/k)$ queries and running time $Time(\hat{\mathcal{A}}) + O(Q')$.

References

1 David J. Aldous and Umesh V. Vazirani. "Go with the winner" algorithms. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 492–501. IEEE Computer Society, 1994. doi:10.1109/SFCS.1994.365742.

2 Sepehr Assadi, Yu Chen, and Sanjeev Khanna. Sublinear algorithms for $(\Delta + 1)$ vertex coloring. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 767–786. SIAM, 2019. doi:10.1137/1.9781611975482.48.

3 Sepehr Assadi and Chen Wang. Sublinear time and space algorithms for correlation clustering via sparse-dense decompositions. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 10:1–10:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ITCS.2022.10.

4 Yossi Azar, Andrei Z. Broder, Anna R. Karlin, Nathan Linial, and Steven Phillips. Biased random walks. In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '92, pages 1–9, New York, NY, USA, 1992. Association for Computing Machinery. doi:10.1145/129712.129713.

5 Amir Azarmehr, Soheil Behnezhad, Alma Ghafari, and Madhu Sudan. Lower bounds for non-adaptive local computation algorithms. In *66th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2025, Sydney, Australia, December 14-17, 2025*, 2025. doi:10.1109/FOCS63196.2025.00078.

6 Soheil Behnezhad. Time-optimal sublinear algorithms for matching and vertex cover. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 873–884. IEEE, 2021. doi:10.1109/FOCS52979.2021.00089.

7 Soheil Behnezhad, Mohammad Roghani, and Aviad Rubinstein. Local computation algorithms for maximum matching: New lower bounds. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 2322–2335, 2023. doi:10.1109/FOCS57990.2023.00143.

8 Soheil Behnezhad, Mohammad Roghani, and Aviad Rubinstein. Sublinear time algorithms and complexity of approximate maximum matching. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 267–280, 2023. doi:10.1145/3564246.3585231.

9 Soheil Behnezhad, Mohammad Roghani, and Aviad Rubinstein. Approximating maximum matching requires almost quadratic time. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 444–454, 2024. doi:10.1145/3618260.3649785.

10 Soheil Behnezhad, Mohammad Roghani, Aviad Rubinstein, and Amin Saberi. Sublinear algorithms for TSP via path covers. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia*, volume 297 of *LIPIcs*, pages 19:1–19:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ICALP.2024.19.

11 Michael A Bender and Dana Ron. Testing properties of directed graphs: acyclicity and connectivity. *Random Structures & Algorithms*, 20(2):184–205, 2002. doi:10.1002/RSA.10023.

12 Sayan Bhattacharya, Peter Kiss, and Thatchaphol Saranurak. Dynamic $(1+\varepsilon)$-approximate matching size in truly sublinear update time. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 1563–1588. IEEE, 2023. doi:10.1109/FOCS57990.2023.00095.

13 Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In F. Frances Yao and Eugene M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 235–244. ACM, 2000. doi:10.1145/335305.335334.

14 Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. Approximating the minimum spanning tree weight in sublinear time. In Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen, editors, *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings*, volume 2076 of *Lecture Notes in Computer Science*, pages 190–200. Springer, 2001. doi:10.1007/3-540-48224-5_16.

15 Yu Chen, Sanjeev Khanna, and Zihan Tan. Query complexity of the metric steiner tree problem. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 4893–4935. SIAM, 2023. doi:10.1137/1.9781611977554.CH179.

16 Yu Chen, Sanjeev Khanna, and Zihan Tan. Sublinear algorithms and lower bounds for estimating MST and TSP cost in general metrics. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPIcs*, pages 37:1–37:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ICALP.2023.37.

17 Artur Czumaj and Christian Sohler. Estimating the weight of metric minimum spanning trees in sublinear-time. In László Babai, editor, *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 175–183. ACM, 2004. doi:10.1145/1007352.1007386.

18 Ioana Dumitriu, Prasad Tetali, and Peter Winkler. On playing golf with two balls. *SIAM J. Discret. Math.*, 16(4):604–615, 2003. doi:10.1137/S0895480102408341.

19 Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 406–415. ACM, 1997. doi:10.1145/258533.258627.

20 Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989. doi:10.1137/0218012.

21 Svante Janson and Yuval Peres. Hitting times for random walks with restarts. *SIAM J. Discret. Math.*, 26(2):537–547, 2012. doi:10.1137/100796352.

22 Slobodan Mitrović, Ronitt Rubinfeld, and Mihir Singhal. Locally Computing Edge Orientations. In *32nd Annual European Symposium on Algorithms (ESA 2024)*, volume 308 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 89:1–89:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ESA.2024.89.

23 Huy N. Nguyen and Krzysztof Onak. Constant-time approximation algorithms via local improvements. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 327–336. IEEE Computer Society, 2008. doi:10.1109/FOCS.2008.81.

24 Uwe Schöning. A probabilistic algorithm for k-sat and constraint satisfaction problems. In *40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA*, pages 410–414. IEEE Computer Society, 1999. doi:10.1109/SFFCS.1999.814612.

25 Vihan Shah. Sublinear-time lower bounds for approximating matching size using non-adaptive queries. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 5027–5065. SIAM, 2026. doi:10.1137/1.9781611978971.182.

26 Yuichi Yoshida, Masaki Yamamoto, and Hiro Ito. An improved constant-time approximation algorithm for maximum matchings. In Michael Mitzenmacher, editor, *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 – June 2, 2009*, pages 225–234. ACM, 2009. doi:10.1145/1536414.1536447.

A Further Connections to Sublinear-Time Graph Algorithms

In this section, we further elaborate on the connections to sublinear-time graph algorithms by giving a concrete example where the lower bound is captured by Markov chains with rewinding.

We examine the lower bound of [11] for testing acyclicity in directed graphs. Given adjacency list access, the algorithm must determine whether the input graph is acyclic or ε-far from acyclic (i.e., at least an ε-fraction of the edges must be removed so that no directed cycles remain in the graph) with a probability of $\frac{2}{3}$. They prove that, for $\varepsilon \leq \frac{1}{16}$, any such algorithm requires at least $\Omega(n^{1/3})$ queries to the adjacency list. We show that this lower bound is captured by our model. First, we examine the case where the algorithm has access

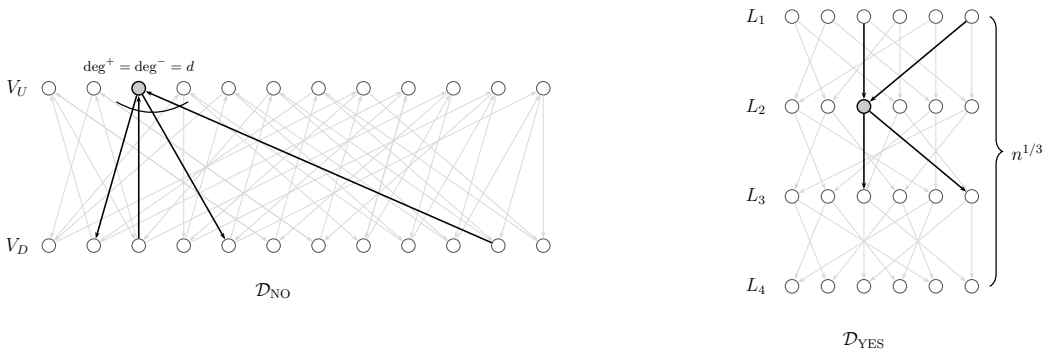


Figure A.1 Sample graphs from the $\mathcal{D}_{\text{YES}}$ (acyclic) and $\mathcal{D}_{\text{NO}}$ (far from acyclic) distributions. The typical vertex has an in-degree and out-degree of d.

only to the outgoing edges (and the out-degree) of a vertex, and later extend it to a setting where the algorithm has access to both the outgoing and incoming adjacency lists at the same time (as well as the in-degree and out-degree).

We start by reviewing the construction of [11]. By Yao's minimax lemma, to prove the lower bound, it suffices to construct two input distributions $\mathcal{D}_{\text{YES}}$ and $\mathcal{D}_{\text{NO}}$, respectively consisting of acyclic and far-from-acyclic graphs, such that any deterministic algorithm with $O(n^{1/3})$ queries fails to distinguish between them with a sufficiently large probability. The construction uses a constant degree-parameter $d \geq 128$ shared between $\mathcal{D}_{\text{YES}}$ and $\mathcal{D}_{\text{NO}}$.

The $\mathcal{D}_{\text{NO}}$ distribution constructs a graph by randomly partitioning the vertices into two groups, V_U and V_D, each of size $n/2$. For the edges, a bipartite d-regular graph between V_U and V_D is chosen uniformly at random and directed towards V_D, and another one is chosen similarly and directed towards V_U. As a result, each vertex in the graph has both in-degree and out-degree equal to d, and the graph is ε-far from acyclic with a high probability of $1 - 2^{-n}$, when $\varepsilon < \frac{1}{16}$ and $d \geq 128$ [11, Lemma 5].

The $\mathcal{D}_{\text{YES}}$ distribution constructs a graph by randomly partitioning the vertices into $n^{1/3}$ groups, $L_1, L_2, \ldots, L_{n^{1/3}}$, each of size $n^{2/3}$. For the edges, for each $1 \leq i < n^{1/3}$, a d-regular graph between L_i and L_{i+1} is chosen uniformly at random and directed towards L_{i+1}. As a result, the graph has no directed cycles and every vertex, except those of L_1 and $L_{n^{1/3}}$, has an in-degree and out-degree equal to d. The distributions are depicted in Figure A.1.

Now, we present the Markov chain that captures this lower-bound construction. The Markov chain consists of two parts: one for $\mathcal{D}_{\text{NO}}$ and one for $\mathcal{D}_{\text{YES}}$. In the NO part, each of the vertex groups V_U and V_D is represented by a state, v_U and v_D respectively. Considering that in the graph, the outgoing edges of V_U go to V_D and the outgoing edges of V_D go to V_U, in the Markov chain, we let v_D transition to v_U, and let v_U transition to v_D with probability 1. To model starting at a random vertex, we use an initial state x_{NO} that transitions to v_D or v_U with probability $\frac{1}{2}$ each. Similarly, in the YES part, each vertex group L_i is represented by a state ℓ_i. Each state ℓ_i for $1 \leq i < n^{1/3}$, transitions to the ℓ_{i+1} with probability 1. To model starting at a random vertex, we use an initial state x_{YES} that transitions to each of $\ell_1, \ell_2 \ldots, \ell_{n^{1/3}}$ with probability $\frac{1}{n^{1/3}}$. As the algorithm interacts with the Markov chain, the only observation it makes is whether the drawn state is $\ell_{n^{1/3}}$ or not. The Markov chain is illustrated in Figure A.2.

Next, we elaborate on how this Markov chain captures the lower bound. The key idea is that an algorithm that makes $O(n^{1/3})$ queries, does not discover any vertex more than once, with a sufficiently high constant probability. That is, the set of edges discovered by the sublinear algorithm does not contain any cycles, even when the directions are ignored. Let $(u_1, i_1), \ldots, (u_q, i_q)$ be the set of queries made to the adjacency lists. We assume without

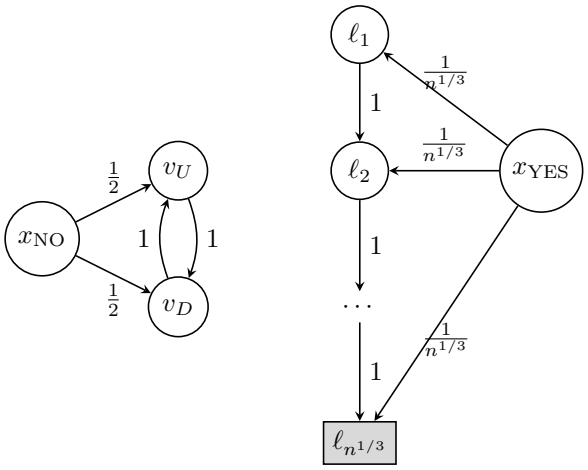


Figure A.2 A Markov chain that models exploring the input graph drawn from the $\mathcal{D}_{\text{YES}}$ and $\mathcal{D}_{\text{NO}}$ distributions. Distinguishing the initial states x_{YES} and x_{NO} is equivalent to distinguishing between the two input distributions.

loss of generality that the queries are unique (i.e., the same query is not made twice), and that the degree of each vertex is revealed to the algorithm as it is discovered.

▶ **Lemma A.1** ([11, Lemma 7]). *Consider any algorithm that makes $q \leq \frac{1}{4} \cdot n^{1/3}$ adjacency-list queries, and let $v_1, \ldots v_q$ be the answers, where $v_k \in V \cup \{\perp\}$. Then, with probability $1 - \frac{1}{16}$, no vertex appears in the answers more than once.*

As a result, we can assume without loss of generality that no vertex appears twice in the answers, since $\frac{1}{16}$ is a negligible probability and when a vertex appears twice, we can simply presume that the algorithm successfully distinguishes between the input distributions. Making this assumption, we argue that distinguishing between the input distributions $\mathcal{D}_{\text{YES}}$ and $\mathcal{D}_{\text{NO}}$ is equivalent to distinguishing between initial states x_{YES} and x_{NO} in the Markov chain. Since there are no repeated vertices, the subgraph explored by the algorithm is a set of rooted trees. When the algorithm explores the neighbors of a vertex u in a group of vertices A, the discovered neighbor v is a random vertex from a neighboring group B. The distribution of B is solely determined by A and does not depend on u.[4] Furthermore, the algorithm observes only the out-degree of each discovered vertex (which is different only for vertices in $L_{n^{1/3}}$), and not its group. This is paralleled by the transition probabilities in the Markov chain, and the partial observation of the state, which only differentiates $\ell_{n^{1/3}}$.

As a result, to prove the lower bound for acyclicity testing, it suffices to prove the following in the context of Markov chains:

▶ **Lemma A.2.** *Any algorithm that distinguishes between the initial states x_{YES} and x_{NO} with a probability of $5/8$, must make $\frac{1}{4} \cdot n^{1/3}$ queries to the Markov chain.*

The proof in [11] essentially argues the same thing, without ever explicitly defining a Markov chain. We give a high-level overview of their proof here. Since the only observation the algorithm can make is whether it has reached $\ell_{n^{1/3}}$ or not, it suffices to bound the probability of reaching $\ell_{n^{1/3}}$ when the initial state is x_{YES}. The next state drawn from x_{YES}

[4] in this construction, B is in fact a deterministic function of A.

is always a state ℓ_i, where $i \in \{1, 2, \ldots, n^{1/3}\}$ is uniformly random. As a result, exploring a tree of depth D from such a state, reaches $\ell_{n^{1/3}}$ with probability of at most $\frac{D+1}{n^{1/3}}$. Therefore, since the sum of the depths of the explored trees is $O(n^{1/3})$, the probability of reaching $\ell_{n^{1/3}}$ throughout the algorithm can be bounded by a constant.

Finally, we show that the lower bound is captured by our model in the more general case, where the algorithm is allowed to query both the incoming and outgoing adjacency lists. To do so, we use a simple gadget. On the YES side of the Markov chain, for each state ℓ_i, we create two paths of length two: A path $\ell_i \to d_i \to \ell_{i+1}$, corresponding to traversing the outgoing edges of L_i, and a path $\ell_i \to u_i \to \ell_{i-1}$, corresponding to traversing the incoming edges of L_i (the NO side is handled similarly). Additionally, we extend the observations made by the algorithm. Other than observing whether the current state is $\ell_{n^{1/3}}$, the algorithm can also observe whether the current state is (1) ℓ_1, the first layer corresponding to vertices with zero in-degree, (2) an auxiliary state d_i, indicating the traversal of an outgoing edge, and (3) an auxiliary state u_i, indicating the traversal of an incoming edge. This allows the algorithm to choose which of the directions it takes, while increasing the number of queries to the Markov chain only by a constant factor. See Section 6 for a discussion of Markov chains with a constant-sized observation alphabet.

One-Way Functions and Polynomial-Time Dimension

Satyadev Nandakumar
Department of Computer Science and Engineering, Indian Institute of Technology Kanpur, India

Subin Pulari
National Research University Higher School of Economics Moscow, Russia

Akhil S
Department of Computer Science and Engineering, Indian Institute of Technology Kanpur, India

Suronjona Sarma
Department of Computer Science and Engineering, Indian Institute of Technology Kanpur, India

Abstract

The theory of randomness in computation has developed around several viewpoints on randomness and information, including statistical tests, compression schemes, betting strategies, and efficiently samplable sources. At the level of computability, several of these viewpoints turn out to be equivalent. A fundamental question is how robust these viewpoints remain when the underlying algorithms are subject to computational resource bounds. This paper studies this question for two polynomial-time notions of information density for infinite binary sequences. Polynomial-time dimension, denoted $\dim_{\mathrm{P}}$, quantifies information density using polynomial-time betting strategies called s-gales. Polynomial-time Kolmogorov complexity rate, denoted $\mathcal{K}_{\mathrm{poly}}$, gives a compression-based notion of polynomial-time information density using polynomial-time descriptions. Hitchcock and Vinodchandran (CCC 2004) showed that $\dim_{\mathrm{P}}(X) \geq \mathcal{K}_{\mathrm{poly}}(X)$ for every sequence X, and asked whether equality always holds. This question was later also posed by Stull.

Our main result proves a duality between the non-robustness of these polynomial-time information-density notions and the existence of one-way functions. Assuming one-way functions exist, we construct a polynomial-time samplable distribution over infinite sequences such that, with probability 1, the sampled sequence X satisfies $\dim_{\mathrm{P}}(X) > \mathcal{K}_{\mathrm{poly}}(X)$ by a uniform gap. Conversely, we show that if some polynomial-time samplable distribution yields such an almost-sure uniform separation, then infinitely-often one-way functions exist. Thus, separations between these two polynomial-time information-density notions over efficiently samplable sources lie at the same frontier as one-way functions, with the reverse direction yielding infinitely-often one-way functions. This duality gives a negative answer, assuming one-way functions, to the open question posed by Hitchcock, Vinodchandran, and Stull. Furthermore, we show that there are individual sequences witnessing separations between $\dim_{\mathrm{P}}$ and $\mathcal{K}_{\mathrm{poly}}$, and that the gap can be made arbitrarily close to 1. We also establish analogous bounds for strong polynomial-time dimension and asymptotic upper polynomial-time Kolmogorov complexity rates. The main technical challenge is to connect finite-length pseudorandomness assumptions with asymptotic information-density measures on infinite sequences. Our proof addresses this by developing several new constructions and arguments involving probabilistic tools such as the Borel–Cantelli lemma, Kolmogorov's inequality for martingales, and techniques for approximating probabilities of efficiently samplable distributions. The work shows that the question of non-robustness for polynomial-time information-density notions, which is prima facie different from standard questions about randomness, pseudorandomness, and cryptography, is in fact intimately related to the existence of one-way functions.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization; Theory of computation → Complexity classes; Theory of computation → Cryptographic primitives

Keywords and phrases Polynomial-time dimension, One-way functions, Resource bounded randomness, Kolmogorov complexity, Polynomial-time martingales

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.44



© Satyadev Nandakumar, Subin Pulari, Akhil S, and Suronjona Sarma;
licensed under Creative Commons License CC-BY 4.0



Category RANDOM

Related Version *ECCC Report*: https://eccc.weizmann.ac.il/report/2025/028/

Funding *Subin Pulari*: Supported by the Basic Research Program at HSE University and this paper was prepared in the framework of this program.
Suronjona Sarma: Supported by the Visvesvaraya PhD Scheme Scholarship. This research work was partially supported by the Research-I Foundation of the Department of Computer Science and Engineering, IIT Kanpur for conference travel.

Acknowledgements The authors would like to thank CS Bhargav, Laurent Bienvenu, John Hitchcock, Elvira Mayordomo, Jaikumar Radhakrishnan, Mahesh Sreekumar Rajasree, Bruno Bauwens and Marius Zimand for helpful comments and discussions.

1 Introduction

An infinite binary sequence can contain regularity that is exposed in different ways. A statistical test may detect a deviation from randomness, a short description may compress many of its prefixes, or a betting strategy may turn predictable patterns into unbounded capital. These perspectives lead both to qualitative notions of randomness and quantitative measures of how much randomness remains visible to a given computational model.

This paper studies one such measure: polynomial-time dimension. We address the longstanding open question of whether its betting and compression formulations continue to agree. We give a negative answer under the assumption that one-way functions exist. The connection is not limited to deriving separations from cryptographic assumptions: we also prove a converse showing that such separations over efficiently samplable sources imply infinitely-often one-way functions.

Constructive dimension is an algorithmic analogue of classical Hausdorff dimension for individual infinite sequences. It was introduced by Lutz [27] via *unfair constructive betting strategies* on infinite sequences, called s-gales. Intuitively, an s-gale reads a sequence bit by bit and repeatedly bets its current capital on the next bit. The parameter s controls the payoff rule of the game. When s is smaller, the game is less favorable to the gambler, so success requires the gambler to exploit stronger regularities in the sequence. The gale succeeds on a sequence if its capital becomes arbitrarily large along that sequence. The constructive dimension of a sequence is the infimum over all values of s for which some constructive s-gale succeeds. Thus, constructive dimension is a threshold measure of how difficult it is for constructive betting strategies to extract profit from the sequence.

Lutz [28] and Mayordomo [29] gave an equivalent characterization of constructive dimension of a sequence in terms of the Kolmogorov complexity of its prefixes. Here $K(x)$ denotes the length of the shortest program that outputs the finite string x. In this formulation, the information density of a sequence is measured by comparing the shortest description lengths of its prefixes with their corresponding lengths. More precisely, this compression view is governed by the lower asymptotic rate of prefix complexity, namely the liminf of $K(X \upharpoonright n)/n$, where $X \upharpoonright n$ denotes the first n bits of X. Prefixes that admit much shorter descriptions indicate lower information density, while prefixes that resist compression indicate higher information density. Thus, for constructive dimension, the betting formulation and the compression formulation define the same notion of information density. These equivalent characterizations establish the robustness of constructive dimension as a mathematical concept. They also provide multiple viewpoints that are useful in different contexts for studying randomness and information density. This leaves a natural robustness problem in the polynomial-time setting: whether the compression characterization continues to match the betting characterization once both are restricted to efficient computation.

Polynomial-time dimension is the analogue of constructive dimension at the level of feasible or polynomial-time computation. As above, there are multiple approaches towards quantifying polynomial-time density of information. The *compressibility* approach uses the time-bounded Kolmogorov complexity of finite prefixes of infinite sequences. The resulting quantity, denoted $\mathcal{K}_{\text{poly}}$, is obtained from the lower asymptotic rate of prefix complexity when the programs producing the prefixes are required to run in polynomial time. In other words, $\mathcal{K}_{\text{poly}}$ measures the information density that remains after compression is restricted to efficient descriptions. The *gambling* approach defines polynomial-time dimension, denoted $\dim_{\text{P}}$, using polynomial-time betting algorithms known as s-gales [27]. Here again, an s-gale succeeds if its capital becomes arbitrarily large along the sequence, but now the gale itself must be computable in polynomial time. The value $\dim_{\text{P}}(X)$ is the infimum over all values of s for which some polynomial-time s-gale succeeds on X. Thus, $\dim_{\text{P}}$ measures the information density that cannot be exploited away by any polynomial-time betting strategy.

At the levels of polynomial-space computation [16] and finite-state computation [5, 4], these notions are known to be equivalent. At the polynomial-time level, Hitchcock and Vinodchandran proved that $\dim_{\text{P}}$ is at least as large as $\mathcal{K}_{\text{poly}}$. The converse direction has remained open for two decades. This question of *robustness of polynomial-time dimension* was posed by Hitchcock and Vinodchandran [14, 16], and later mentioned in a survey by Stull (see Open Question 3.71 in [36]).

The positive equivalences at the computability and polynomial-space levels rely on computational power that can effectively search over descriptions or simulate betting strategies. Such conversions are not available at polynomial time. Thus, in the polynomial-time setting, the robustness problem becomes sensitive to the computational limits imposed on the compression and betting algorithms. A related positive characterization is known for polynomial-time dimension when the compression side is formulated using reversible polynomial-time compressors that do not start from scratch [26]. In contrast, the question addressed in this paper concerns the original time-bounded Kolmogorov-complexity formulation of Hitchcock and Vinodchandran.

In this work, we give a surprising negative resolution to this question under the standard cryptographic assumption that one-way functions exist. First, assuming one-way functions, we construct a polynomial-time samplable distribution ν over Σ^∞ for which a set of sequences of ν-measure 1 exhibits a strict dimension gap between $\dim_{\text{P}}$ and $\mathcal{K}_{\text{poly}}$. We then strengthen this from sets to individual sequences, showing that there are sequences for which $\dim_{\text{P}} \neq \mathcal{K}_{\text{poly}}$ under the same assumption. Moreover, the gap $\dim_{\text{P}} - \mathcal{K}_{\text{poly}}$ can be made arbitrarily close to 1. The analogous non-robustness also holds for *strong* polynomial-time dimension, where lower asymptotic rates are replaced by upper asymptotic rates.

We also prove a partial converse: the existence of a polynomial-time samplable distribution supporting such dimension gaps implies the existence of infinitely-often one-way functions. This shows that the connection is not limited to deriving dimension gaps from cryptographic assumptions. Via the converse implication, dimension gaps over efficiently samplable sources lead to infinitely-often one-way functions.

Formally connecting one-way functions to the failure of robustness for these asymptotic measures of randomness requires arguments tailored to infinite sequences. The classical connections between compressibility, indistinguishability, and pseudorandomness in the finite-string setting [17, 38, 21] do not by themselves control information rates along infinitely many prefixes, or the success of a betting strategy along an infinite sequence.

The forward implication has to pass from asymptotic betting success on infinite sequences to finite-length distinguishers. A key step is an argument involving the Borel–Cantelli lemma, which isolates infinitely many lengths where the constructed source produces prefixes

on which the betting strategy has substantial gain. This is paired with a martingale estimate based on Kolmogorov's inequality, showing that comparable gains are unlikely under uniform randomness. The separation between these two behaviors is then turned into a pseudorandomness distinguisher.

The converse direction is technically different. Here the task is to show that an almost-sure dimension gap over an efficiently samplable source already carries cryptographic content. Assuming that infinitely-often one-way functions do not exist, one can approximate the probabilities of samplable distributions on typical inputs. The proof has to assemble these finite-prefix approximations into an efficient betting algorithm that works along the infinite sequence. This converts the absence of cryptographic hardness into a robustness statement for polynomial-time information density, and hence forces any such dimension gap to imply infinitely-often one-way functions.

In addition to resolving a longstanding open problem under the assumption that one-way functions exist, our results show that the robustness of polynomial-time information density is not merely a technical question internal to resource-bounded dimension. It is governed by the same phenomenon that underlies modern cryptography: the existence of efficiently samplable objects whose structure cannot be uniformly exposed by feasible computation. Thus the paper establishes a direct bridge between one-way functions and asymptotic information measures.

2 Background: Betting and Compression

Notation. Let Σ denote the binary alphabet $\{0,1\}$. Σ^n denotes the set of n-length strings and Σ^* denotes the set of all finite strings over Σ. For a finite string $w \in \Sigma^*$, $|w|$ denotes the length of w. 1^* denotes the set of all finite strings in the unary alphabet $\{1\}$. Σ^∞ denotes the *Cantor space*, or the set of all infinite strings over Σ. We use small letters w, x to represent finite strings and capital letters X, Y to represent infinite strings over Σ. For an infinite string $X \in \Sigma^\infty$, $X[i]$ denotes the ith bit of X and $X \upharpoonright n$ denotes the first n bits of X. For infinite string $X = X_1X_2X_3X_4\ldots$ and $m \geq n$, $X[n:m]$ denotes the the substring $X_nX_{n+1}X_{n+2}\ldots X_m$. For a finite string $w \in \Sigma^*$, $w[i]$, $w \upharpoonright n$ and $w[n:m]$ are defined similarly. λ denotes the empty string. $\mathbb{N}$ denotes the set of natural numbers and $\mathbb{Q}$ denotes the set of rationals. All the logarithms in the paper are taken to base 2 unless specified otherwise. For any function $f : \Sigma^\infty \to \Sigma^\infty$ and $\mathcal{S} \subseteq \Sigma^\infty$ let $f(\mathcal{S})$ denote $\{f(X) : X \in \mathcal{S}\}$.

For $w \in \Sigma^*$, C_w denotes the *cylinder set* of w, that is the set of all infinite sequences in Σ^∞ that begin with w. Let $\mathcal{B}(\Sigma^\infty)$ denote the Borel σ-algebra over Σ^∞ generated by the set of all cylinder sets $\{C_w : w \in \Sigma^*\}$. For two finite strings $x, y \in \Sigma^*$, $x.y$ denotes the concatenation of the strings x and y.

We use $\text{poly}(n)$ to denote the set of polynomial (time) functions $\bigcup_{c\in\mathbb{N}}\{n^c\}$. When the context is clear, we also use poly to denote $\text{poly}(n)$. We use U_n to denote the uniform probability distribution over Σ^n.

2.1 s-gales and Kolmogorov Complexity

***s*-gales** are real-valued betting strategies on infinite sequences. An s-gale may be viewed as a gambler that reads a sequence bit by bit and repeatedly reallocates its current capital between the two possible next bits. The gale d starts with initial capital 1 on the empty string. At the nth stage, after seeing a prefix $X_1X_2\ldots X_n$, the gale has capital $d(X_1X_2\ldots X_n)$ and places a bet on the next bit X_{n+1}. The rule of the game is that, for every finite string w, the capital assigned to the two possible extensions satisfies $2^s d(w) = d(w0) + d(w1)$. A gale d succeeds on the sequence $X \in \Sigma^\infty$ if it can make its capital arbitrarily large along the sequence, more precisely if $\limsup_{n\to\infty} d(X_1X_2\ldots X_n) = \infty$.

▶ **Definition 1** ([27, 28]). *For $s \in [0,\infty)$, an* s-supergale *is a function $d : \Sigma^* \to [0,\infty)$ such that $d(\lambda) < \infty$ and for every finite string $w \in \Sigma^*$, $d(w) \geq 2^{-s}(d(w0) + d(w1))$. An* s-gale *is a function $d : \Sigma^* \to [0,\infty)$ with $d(\lambda) < \infty$ for every finite string $w \in \Sigma^*$, $d(w) = 2^{-s}(d(w0) + d(w1))$. A* supermartingale *is a 1-supergale. A* martingale *is a 1-gale.*

Unless specified otherwise, for every s-supergale d, we assume that the initial capital $d(\lambda) = 1$.

▶ **Definition 2** ([27, 28],[1]). *The* success set *of an s-supergale d is $S^\infty(d) = \{ X \in \Sigma^\infty \mid \limsup_{n\to\infty} d(X \upharpoonright n) = \infty \}$. The* strong success set *of an s-supergale d is $S^\infty_{\text{str}}(d) = \{ X \in \Sigma^\infty \mid \liminf_{n\to\infty} d(X \upharpoonright n) = \infty \}$.*

For example, fix $s > 0$. The strategy that always bets all its capital on the next bit being 0 is given by $d(\lambda) = 1$, $d(w0) = 2^s d(w)$, and $d(w1) = 0$. This satisfies the s-gale condition. Along the sequence $000\cdots$, its capital after n steps is 2^{sn}, and hence it succeeds. On the other hand, a single bit 1 makes the capital drop to 0.

A more interesting example is obtained by betting according to a fixed bias. Let $0 < p < 1$, and define

$$d(w) = 2^{s|w|} p^{\#_0(w)} (1-p)^{\#_1(w)},$$

where $\#_0(w)$ and $\#_1(w)$ are the numbers of zeros and ones in w. Then $d(w0)+d(w1) = 2^s d(w)$, so d is an s-gale. If a sequence has limiting frequency p of zeros and $1-p$ of ones, then along its prefixes the exponential growth rate of this capital is $s - H(p)$, where $H(p)$ is the binary entropy. Thus, for every $s > H(p)$, this gale succeeds on such a biased sequence. This example illustrates how an s-gale can exploit statistical regularity rather than merely a fixed deterministic pattern.

When $s = 1$, the average capital after betting on $w0$ or $w1$ is equal to the previous capital $d(w)$. In this case the betting strategy is fair, and such s-gales are precisely martingales. When $s < 1$, the average capital after the bet is strictly smaller than $d(w)$. Thus the gambler is playing in an unfavorable market. This is often interpreted as "betting under inflation": unless the gambler detects structure in the sequence and bets accordingly, its capital tends to shrink. As s decreases, success requires increasingly strong regularities in the sequence.

The **Kolmogorov complexity** of a finite string x, denoted by $K(x)$, is the length of the shortest program that outputs x. That is, $K(x) = \min\{|\pi| : \mathcal{U}(\pi) = x\}$, where π is a program and $\mathcal{U}$ is the universal Turing machine. Any string x can be produced by a program that simply prints x, and therefore $K(x) \leq |x| + O(1)$. However, if the string contains regularity, there may be a much shorter program that outputs it. For instance, if all the even bits of the string are 0, then the program only needs to encode the bits at the odd indices and insert zeros in the even positions. In that case, $K(x)$ is at most about half the length of the string, up to additive constants.

▶ **Definition 3** (t-time-bounded Kolmogorov complexity [22, 24]). *Let t be a time-constructible function, and $x \in \Sigma^*$ be a finite binary string. The t-time bounded Kolmogorov complexity of x is $K_t(x) = \min\{|\Pi| \mid \Pi \in \Sigma^*$ and $\mathcal{U}(\Pi) = x$ in $t(|x|)$ steps$\}$, where $\mathcal{U}$ is any fixed efficient universal Turing machine.*

Since in the study of resource-bounded dimension we divide $K_t(x)$ by $|x|$, our results are unaffected by the choice between prefix-free Kolmogorov complexity and plain Kolmogorov complexity. We use plain Kolmogorov complexity in the rest of the paper.

2.2 Polynomial-time Dimension

Resource-bounded dimension is defined by placing resource bounds on the computation of the s-gale. Analogously, resource-bounded Kolmogorov complexity is defined by placing resource bounds on the programs that print a string. **Polynomial-time dimension** quantifies the rate of information in an infinite sequence, measured with respect to polynomial-time bounded computation. It is formulated using polynomial-time s-gales. An s-gale d wins on an infinite sequence X if $\limsup_{n\to\infty} d(X \upharpoonright n) = \infty$. The gale d is said to be polynomial-time computable if, for some $p(n) \in \mathrm{poly}(n)$, the value $d(w)$ can be computed in at most $p(n)$ time, where n is the length of w. Intuitively, $\dim_{\mathrm{P}}(X)$ is the infimum of the values of s for which there exists a polynomial-time s-gale that succeeds on X. Smaller values of s correspond to a more unfavorable betting environment. Thus, a low polynomial-time dimension means that some efficient betting strategy can exploit substantial regularity in the sequence, whereas high polynomial-time dimension means that the sequence resists such efficient betting.

On the other hand, in the *incompressibility approach*, we use the *time-bounded Kolmogorov complexity* to define a polynomial-time analogue of $\mathcal{K}$. For a given time bound $t(n)$, the ***t*-time bounded Kolmogorov complexity** $K_t(x)$ of a string x is the length of the shortest program that generates x in at most $t(|x|)$ time steps (see [24]).

For a given time bound $t(n)$, the quantity $\liminf_{n\to\infty} K_t(X \upharpoonright n)/n$ gives the asymptotic t-time-bounded rate of information in an infinite string X. The polynomial-time analogue of $\mathcal{K}$ is the infimum of this quantity over all polynomial-time bounds t.

Now, we define the asymptotic polynomial-time density of information of a subset $\mathcal{F} \subseteq \Sigma^\infty$.

▶ **Definition 4** ($\mathcal{K}_{\mathrm{poly}}$ and $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}$ [16, 36, 15]). *For any $\mathcal{F} \subseteq \Sigma^\infty$, define $\mathcal{K}_{\mathrm{poly}}(\mathcal{F}) = \inf_{t\in\mathrm{poly}} \sup_{X\in\mathcal{F}} \liminf_{n\to\infty} \frac{K_t(X\upharpoonright n)}{n}$ and $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(\mathcal{F}) = \inf_{t\in\mathrm{poly}} \sup_{X\in\mathcal{F}} \limsup_{n\to\infty} \frac{K_t(X\upharpoonright n)}{n}$.*

For individual sequences, we define $\mathcal{K}_{\mathrm{poly}}(X) = \mathcal{K}_{\mathrm{poly}}(\{X\})$ and $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(X) = \mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(\{X\})$. Note that $\mathcal{K}_{\mathrm{poly}}$ and $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}$ are equivalently defined in the following way.

▶ **Definition 5** ($\mathcal{K}_{\mathrm{poly}}$ and $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}$ [16, 36, 15]). *For any $X \in \Sigma^\infty$, define $\mathcal{K}_{\mathrm{poly}}(X) = \inf_{t\in\mathrm{poly}} \liminf_{n\to\infty} \frac{K_t(X\upharpoonright n)}{n}$ and $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(X) = \inf_{t\in\mathrm{poly}} \limsup_{n\to\infty} \frac{K_t(X\upharpoonright n)}{n}$.*

Now, we define time-bounded supergales. We first define $t(n)$-time computable s-gales.

▶ **Definition 6** ($t(n)$-time computability [27]). *An s-supergale $d : \Sigma^* \to [0,\infty)\cap\mathbb{Q}$ is $t(n)$-time computable if there exist a $O(t(n))$-time computable function $f : \Sigma^* \times 1^* \to \mathbb{Q}$ such that $|d(w) - f(w, 1^n)| \leq 2^{-n}$ for every $w \in \Sigma^*$ and $n \in \mathbb{N}$.*

Another notion of time-bounded computability is that of exact $t(n)$-time computability.

▶ **Definition 7** (Exact $t(n)$-time computability [27]). *An s-supergale $d : \Sigma^* \to [0,\infty) \cap \mathbb{Q}$ is exact $t(n)$-time computable if there exist a $O(t(n))$-time computable function $f : \Sigma^* \to \mathbb{Q}$ such that $d(w) = f(w)$ for every $w \in \Sigma^*$.*

Now, we define polynomial-time dimension and strong dimension.

▶ **Definition 8** (Polynomial-time dimension [27] and strong dimension [1, 36]). *The polynomial-time dimension of $\mathcal{F} \subseteq \Sigma^\infty$ is defined as*

$$\dim_{\mathrm{P}}(\mathcal{F}) = \inf\{s \mid (\exists k) \text{ there is an exact } n^k\text{-time computable } s\text{-gale } d \text{ such that } \mathcal{F} \subseteq S^\infty(d)\}.$$

For a sequence $X \in \Sigma^\infty$*, define* $\dim_{\mathrm{P}}(X) = \dim_{\mathrm{P}}(\{X\})$*. The* polynomial-time strong dimension $\mathrm{Dim}_{\mathrm{P}}(\mathcal{F})$ *of* $\mathcal{F} \subseteq \Sigma^\infty$ *is defined by replacing* S^∞ *by* S^∞_{str} *in the above expression. For a sequence* $X \in \Sigma^\infty$*, define* $\mathrm{Dim}_{\mathrm{P}}(X) = \mathrm{Dim}_{\mathrm{P}}(\{X\})$.

We now make an important remark. Polynomial-time dimension and strong dimension are often defined in terms of n^k-time computable s-gales that need not be exact computable. But these definitions are in fact equivalent. This is a direct consequence of the following lemma from [27] which is also useful in proving the main results in this paper.

▶ **Lemma 9** (Exact Computation Lemma [27]). *If there is a polynomial-time computable s-gale d such that $2^s \in \mathbb{Q}$, then there exists an exact polynomial-time computable s-gale $\tilde{d}$ such that $S^\infty(d) \subseteq S^\infty(\tilde{d})$.*

We will often use the terminology polynomial-time computable gales to refer to exact $t(n)$-time computable s-gales where $t(n) \in \mathrm{poly}(n)$. Similar comment applies in the case of exact polynomial-time computability and other notions related to gales such as martingales and supergales.

2.3 Robustness of Dimension

At the computability level, the relevant notions are $\mathcal{K}$, defined using time-unbounded Kolmogorov complexity $K(x)$, and dim, defined using lower semicomputable s-gales. Lutz [28] and Mayordomo [29] showed that the unbounded notions of $\mathcal{K}$ and dim are equivalent, hence establishing the robustness of *constructive* dimension.

PSPACE analogues of dim and $\mathcal{K}$ are defined by restricting the s-gales in the definition of dim to be polynomial-space computable and by using polynomial-space bounded Kolmogorov complexity in the definition of $\mathcal{K}$. Hitchcock and Vinodchandran [14, 16] showed that these notions of resource-bounded dimension coincide in the PSPACE setting. Thus polynomial-space dimension is robust in the same sense.

This leads to the following question: *Are the notions of polynomial-time dimension formulated using s-gales ($\dim_{\mathrm{P}}$) and time-bounded Kolmogorov complexity ($\mathcal{K}_{\mathrm{poly}}$) equivalent?* Hitchcock and Vinodchandran [16] showed that $\dim_{\mathrm{P}}$ is always greater than or equal to $\mathcal{K}_{\mathrm{poly}}$.

▶ **Theorem 10** ([14, 16]). *For every $\mathcal{F} \subseteq \Sigma^\infty$, $\mathcal{K}_{\mathrm{poly}}(\mathcal{F}) \leq \dim_{\mathrm{P}}(\mathcal{F})$.*

The question whether the reverse inequality holds has remained elusive. Hitchcock and Vinodchandran [14, 16] and later Stull in [36] posed the following open question:

▶ **Question 11** ([14, 16, 36]). *Is it true that, for every sequence $X \in \Sigma^\infty$, $\dim_{\mathrm{P}}(X) = \mathcal{K}_{\mathrm{poly}}(X)$?*

Polynomial-time strong dimension is a dual notion of polynomial-time dimension [1, 36]. As in the case of polynomial-time dimension, there exist two notions, one defined using s-gales and the other defined in terms of time-bounded Kolmogorov complexity. An s-gale d strongly succeeds on an infinite sequence X if $\liminf_{n\to\infty} d(X_1X_2\ldots X_n) = \infty$.

The strong dimension analogue of $\mathcal{K}_{\mathrm{poly}}$ is defined by replacing the lim inf in the definition of $\mathcal{K}_{\mathrm{poly}}$ with lim sup. The analogue of Theorem 10 also holds in this setting: $\mathrm{Dim}_{\mathrm{P}}$ is at least as large as $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}$. Stull in [36] posed the following question for the dual notion of strong dimension:

▶ **Question 12.** *Are the notions of polynomial-time strong dimension formulated using s-gales ($\mathrm{Dim}_{\mathrm{P}}$) and time-bounded Kolmogorov complexity ($\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}$) equivalent for every sequence $X \in \Sigma^\infty$?*

We answer both questions in the negative, under the standard cryptographic assumption that one-way functions exist.

2.4 Measure over Cantor Space

▶ **Definition 13** ([34]). *A function $\mu : \mathcal{M} \to [0, \infty]$, where $\mathcal{M} \subseteq P(\Sigma^\infty)$, is called a measure over $(\Sigma^\infty, \mathcal{M})$ if $\mu(\emptyset) = 0$ and μ is countably additive. That is, for any countable disjoint collection $\{E_k\}_{k=1}^\infty$ of measurable sets, $\mu\left(\bigcup_{k=1}^\infty E_k\right) = \sum_{k=1}^\infty \mu(E_k)$.*

We take $\mathcal{M}$ as the set of Borel Measurable sets. Note that $\mathcal{B} \subseteq \mathcal{M}$, where $\mathcal{B}$ is the Borel sigma algebra over Σ^∞, we omit $\mathcal{M}$ and say that μ is a measure over Σ^∞.

▶ **Definition 14.** *Let ν be a measure over Σ^∞. We say that ν is a probability distribution over Σ^∞ if it holds that $\nu(\Sigma^\infty) = 1$.*

Let ν be a probability distribution over Σ^∞. For every n, let ν_n denote the probability distribution induced by ν on Σ^n defined as $\nu_n(w) = \nu(C_w)$ for every $w \in \Sigma^*$.

Let Σ^∞ be the Cantor space. Now we define the uniform (Lebesgue) measure over the Cantor space [7, 31]. The uniform measure of a cylinder set C_w determined by a finite string w is defined as: $\mu(C_w) = 2^{-|w|}$. Using routine measure theoretic arguments, μ uniquely extends to a probability measure over $(\Sigma^\infty, \mathcal{B}(\Sigma^\infty))$ (see [2]).

2.5 One-way Functions and Pseudorandom Generators

One-way functions [6, 10, 23] are functions that are easy to compute but are hard to invert, except possibly on a negligible fraction of inputs. They are central in cryptography, since their existence is necessary for that of essential cryptographic primitives [17, 33, 12, 8, 30].

The following is the definition of one-way functions secure against uniform probabilistic polynomial-time adversaries.

▶ **Definition 15** (One-way functions [11, 37, 10]). *A function $f_n : \Sigma^* \to \Sigma^*$ is a one-way function (OWF) if there is a constant b such that f_n is computable in time n^b for sufficiently large n, and for every probabilistic polynomial-time algorithm A and every constant $c > 0$, $\Pr[A(f_n(U_n)) \in f_n^{-1}(f_n(U_n))] \leq \frac{1}{n^c}$, for all sufficiently large n, where the probability is taken over the choice of U_n and the internal randomness of algorithm A.*

To define pseudorandom generators, we first define computational indistinguishability.

▶ **Definition 16** (Computational Indistinguishability [11, 37]). *The ensembles of random variables $\{X_n\}_{n\in\mathbb{N}}$ and $\{Y_n\}_{n\in\mathbb{N}}$ are computationally indistinguishable if for every probabilistic polynomial-time algorithm T and every constant $c > 0$,*

$$|\Pr[T(X_n) = 1] - \Pr[T(Y_n) = 1]| \leq \frac{1}{n^c}$$

where the probabilities are taken over the distributions of either X_n or Y_n and the internal randomness of the algorithm T.

▶ **Definition 17** (Pseudorandom generators [11, 37, 10]). *A sequence of functions $\{G_n : \Sigma^{d(n)} \to \Sigma^n\}_{n\in\mathbb{N}}$ is a pseudorandom generator (PRG) if $d(n) < n$ for all n, there exists a uniform deterministic polynomial-time algorithm M such that $M(n, x) = G_n(x)$, and the ensembles $\{G_n(U_{d(n)})\}_{n\in\mathbb{N}}$ and $\{U_n\}_{n\in\mathbb{N}}$ are computationally indistinguishable.*

The theorem below states the equivalence of one-way functions and pseudorandom generators.

▶ **Theorem 18** (Equivalence theorem [17, 11, 10]). *The following are equivalent.*

1. *One-way functions exist.*
2. *There exist a pseudorandom generator with seed length $d(m) = m - 1$.*
3. *For every polynomial-time computable constant $\epsilon \in (0, 1)$, there exists a pseudorandom generator with seed length $d(m) = \epsilon \cdot m$.*

2.6 Efficiently Samplable Sources and Approximable Gales

We generalize the notion of polynomial-time samplable distributions [3] to probability distributions on the space of infinite sequences. This is the distributional setting in which our main duality theorem is stated: instead of considering arbitrary sequences alone, we also study typical outputs of efficiently samplable sources over Σ^∞.

Let ν be a probability measure over Σ^∞. For every n, let $\nu_n(w) = \nu(X \in \Sigma^\infty : w \sqsubseteq X)$[1] denote the probability distribution induced by ν on Σ^n. Now, we define polynomial-time samplable distributions on Σ^∞.

▶ **Definition 19.** *A probability distribution ν over Σ^∞ is polynomial-time samplable if there exists a probabilistic Turing machine M that uses $q(n)$ random bits, where q is a polynomial, such that for every n and $w \in \Sigma^n$, $\Pr_{r \sim \Sigma^{q(n)}}[M(1^n, r) = w] = \nu_n(w)$.*

An alternate way of defining polynomial-time samplable distributions on Σ^∞ is the following: $\nu = \{\nu_n\}$ is a *polynomial-time samplable distribution on* Σ^∞ if $\{\nu_n\}$ is a *polynomial-time samplable distribution* ([3]) such that for any $w \in \Sigma^n$, $\nu_n(w) = \nu_{n+1}(w.0) + \nu_{n+1}(w.1)$. The equivalence of these notions follows using routine measure theoretic arguments.

Primitives like one-way functions and pseudorandom generators are defined against *probabilistic* polynomial-time adversaries, whereas polynomial-time s-gales are *deterministic* betting strategies. To compare these two kinds of objects, we use a mild average-case evaluation notion for gales. The betting strategy itself remains a deterministic s-supergale, with the usual success condition along infinite sequences. The relaxation is only in how its capital function is accessed: on inputs drawn from the target samplable distribution, the capital is required to have an efficient randomized constant-factor approximation with high probability.

This is the natural interface for our distributional arguments. On the cryptographic side, pseudorandomness and probability-estimation results are inherently probabilistic and average-case over samples from a samplable distribution. On the dimension side, the object being analyzed is still a gale. Exact polynomial-time computable gales are included as a special case, since their values can be returned deterministically. Thus the notion below does not replace polynomial-time dimension. It is a technical strengthening used to formulate the almost-sure distributional separation and its converse.

This is in the spirit of the probabilistic approximation notions for martingales studied by Regan and Sivakumar [32], but we only need an inverse-polynomial average-case guarantee over ν_n: for each length n and every constant k, the estimator approximates $d(w)$ within a fixed constant factor on all but an n^{-k} fraction of samples $w \sim \nu_n$, and any approximation errors are allowed only on strings in support of ν_n.

▶ **Definition 20.** *Let $d : \Sigma^* \to [0, \infty) \cap \mathbb{Q}$ be an s-supergale and ν be any probability distribution over Σ^∞. d is $t(n)$-time ν-approximable if for every constant k there exist a probabilistic $t(n)$-time machine M and constant $c < 1$ such that for every n, $\{w \in \Sigma^n : M(w) \notin [c \cdot d(w), d(w)]\} \subseteq \mathrm{supp}(\nu_n)$ and $\nu_n\{w \in \Sigma^n : M(w) \notin [c \cdot d(w), d(w)]\} \leq n^{-k}$.*

[1] $\sqsubseteq$ denotes the prefix operator

Here $\mathrm{supp}(\nu_n) = \{w \in \Sigma^n : \nu_n(w) > 0\}$. The support condition prevents the approximation procedure from hiding errors on impossible prefixes. Equivalently, outside the support of the sampled prefix distribution, the estimator must still output a value within the specified multiplicative range, while on typical samples from the source it may fail only with inverse-polynomial probability.

3 Our Results

3.1 One-way functions and dimension gaps

Our main result gives a two-sided connection between one-way functions and dimension gaps over efficiently samplable sources. Assuming one-way functions, we construct a polynomial-time samplable distribution over Σ^∞ whose typical sequences have low polynomial-time Kolmogorov complexity rate but cannot be captured by low-dimensional polynomial-time betting strategies. Conversely, the existence of such a samplable source already implies infinitely-often one-way functions.

▶ **Theorem 21.** *In the following, (1) $\Longrightarrow$ (2) $\Longrightarrow$ (3):*

1. *One-way functions exist.*
2. *For every $s < 1/2$, there exists a polynomial-time samplable distribution ν over Σ^∞ such that:*
   a. *$\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(X) \le s$ for almost every $X \sim \nu$.*
   b. *For every $s' \in (s, 1/2)$ and every polynomial-time ν-approximable s'-supergale d, we have $\nu(S^\infty(d)) = 0$.*
3. *Infinitely-often one-way functions exist.*

Item (2a) says that, with ν-probability 1, the sampled sequence has small polynomial-time Kolmogorov complexity rate. In particular, $\mathcal{K}_{\mathrm{poly}}(X) \le s$ for ν-almost every $X \sim \nu$. The second part of Item (2) rules out success by low-dimensional polynomial-time ν-approximable supergales. Lemma 42 shows that, under the same assumption, this yields $\dim_{\mathrm{P}}(X) \ge 1/2$ for ν-almost every X. Since $s < 1/2$, this gives an almost-sure dimension gap over the samplable source.

It is important that the approximation notion in Item (2b) does not change the polynomial-time dimension appearing in the applications below. Every exact polynomial-time computable gale is polynomial-time ν-approximable, by returning its value deterministically. Thus Item (2b) rules out, in particular, ordinary polynomial-time betting strategies on positive-measure subsets of the samplable source. The approximability condition is used only to state the distributional theorem in a form compatible with pseudorandomness arguments and average-case probability-estimation results.

We prove Theorem 21 in two steps. First we show that one-way functions yield a polynomial-time samplable distribution exhibiting an almost-everywhere dimension gap between $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}$ and $\dim_{\mathrm{P}}$ (Lemma 25, giving (1)⇒(2)), and then we show that the existence of such a dimension-gap distribution yields infinitely-often one-way functions (Lemma 36, giving (2)⇒(3)).

3.2 Applications

Theorem 21 resolves the original robustness question for individual sequences, assuming one-way functions.

▶ **Theorem 22.** *If for all $X \in \Sigma^\infty$, $\dim_{\mathrm{P}}(X) = \mathcal{K}_{\mathrm{poly}}(X)$, then no one-way functions exist.*

Equivalently, assuming one-way functions, there are individual sequences X such that $\mathcal{K}_{\text{poly}}(X) < \dim_{\text{P}}(X)$. We extend this result to show that if one-way functions exist, then the distance between these quantities can be arbitrarily close to the maximum possible value of 1.

▶ **Theorem 23.** *If one-way functions exist, then for any $\epsilon > 0$, there exists $X \in \Sigma^\infty$ such that $\dim_{\text{P}}(X) - \mathcal{K}_{\text{poly}}(X) \geq 1 - \epsilon$.*

For polynomial-time strong dimension, we show that if one-way functions exist, then there exist sequences for which the gap between $\mathcal{K}^{\text{str}}_{\text{poly}}$ and Dim_{P} is arbitrarily close to 1.

▶ **Theorem 24.** *If one-way functions exist, then for any $\epsilon > 0$, there exists $X \in \Sigma^\infty$ such that $\text{Dim}_{\text{P}}(X) - \mathcal{K}^{\text{str}}_{\text{poly}}(X) \geq 1 - \epsilon$.*

Therefore, we show that if one-way functions exist, the answers to both parts of Open Question 3.71 from [36] are negative.

4 Proof Outline and Techniques

In this section we describe the main constructions and reductions behind our results. The emphasis is on how finite-length pseudorandomness arguments interact with asymptotic information-density notions over infinite sequences.

The main difficulty is that the two sides are formally quite different. Pseudorandomness is a finite-length indistinguishability notion against probabilistic polynomial-time tests, while dimension is an asymptotic notion defined by success of betting strategies along infinitely many prefixes of one sequence. The proof has to convert information in both directions: from asymptotic betting success to finite distinguishers, and from average-case probability estimation for samplable distributions to betting objects on infinite sequences.

4.1 From one-way functions to dimension gaps

We first show how one-way functions, through pseudorandom generators, yield efficiently samplable sources with an almost-everywhere dimension gap. Concretely, assuming one-way functions, for every $s < 1/2$ we construct a polynomial-time samplable distribution ν over Σ^∞ such that (i) $\mathcal{K}^{\text{str}}_{\text{poly}}(X) \leq s$ for ν-almost every X, and (ii) for every $s' \in (s, 1/2)$, no polynomial-time ν-approximable s'-supergale succeeds on a set of positive ν-measure.

▶ **Lemma 25.** *If one-way functions exist, then for every $s < 1/2$, there exist a polynomial-time samplable distribution ν over Σ^∞ such that:*

1. *$\mathcal{K}^{\text{str}}_{\text{poly}}(X) \leq s$ for almost every $X \sim \nu$.*
2. *For every $s' \in (s, 1/2)$ and any polynomial-time ν-approximable s'-supergale d, $\nu(S^\infty(d)) = 0$.*

Fix $s < 1/2$. Assume one-way functions exist, and choose a dyadic parameter $s_0 = 2^{-m}$ with $2s_0 < s$. Then there are pseudorandom generators $\{G_n\}_{n\in\mathbb{N}}$ running in time $t_G(n) \in \text{poly}(n)$ with constant stretch: for each n, $G_n : \Sigma^{s_0 n} \to \Sigma^n$. The choice of such an s_0 will give the stronger compression bound needed to obtain $\mathcal{K}^{\text{str}}_{\text{poly}}(X) \leq s$.

4.1.1 Construction of the PRG-induced source

We extend the family $\{G_n\}$ to a mapping on infinite sequences, $g : \Sigma^\infty \to \Sigma^\infty$, by applying the appropriate G_{2^i} blockwise to successive blocks of the input. Given a PRG $G_n : \Sigma^{s_0 n} \to \Sigma^n$ where $s_0 = 2^{-m}$ for some $m \in \mathbb{N}$, we construct a mapping g from the Cantor set to itself, by

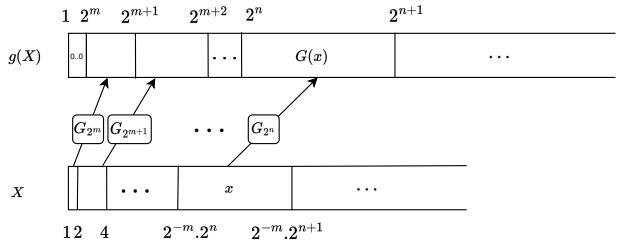


Figure 1 Illustration of the construction of $g(X)$. The first 2^m bits of $g(X)$ are 0. Thereafter, for every $n > m$, the block $X[s_0 \cdot 2^{n-1}, s_0 \cdot 2^n - 1]$ is mapped to $G_{2^{n-1}}(X[s_0 \cdot 2^{n-1}, s_0 \cdot 2^n - 1])$ in the output string $g(X)$.

using the family $\{G_n\}_{n\in\mathbb{N}}$ to map blocks within the input string to *longer* blocks within the output string. Given an input $X \in \Sigma^\infty$, $g(X)$ is constructed by applying G_n to successive blocks x of size $s_0 \cdot 2^n$ from X for all $n \in \mathbb{N}$. We first define a sequence of mappings g_k from Σ^k to Σ^*. These mappings preserve the prefix ordering among finite strings and hence we define $g(X)$ for $X \in \Sigma^\infty$ as the limit of $g_k(X \upharpoonright k)$.

▶ **Definition 26.** *Let $G_n : \Sigma^{s_0 n} \to \Sigma^n$ where $s_0 = 2^{-m}$ for some $m \in \mathbb{N}$ and let $k \in \mathbb{N}$. For $w \in \Sigma^k$, define $g_k(w)$ as follows. For any n satisfying any $m < n \leq \lfloor \log(k/s_0) \rfloor$ and $i \in \{2^{n-1}+1, 2^{n-1}+2, \ldots 2^n\}$, let $g_k(w)[i] = G_{2^{n-1}}(x)[i-2^{n-1}]$, where $x = w[s_0 \cdot 2^{n-1}, s_0 \cdot 2^n - 1]$. For $i \in \{1, 2, \ldots, 2^m\}$, we set $g_k(w)[i] = 0$.*

Note that $g_k(w)[2^{n-1}+1, 2^n] = G_{2^{n-1}}(w[s_0 \cdot 2^{n-1}, s_0 \cdot 2^n - 1])$. For $k \leq 2^m$, g_k maps any $w \in \Sigma^k$ to 0^{2^m}. Observe that the sequence of mappings $\langle g_k \rangle_{k\in\mathbb{N}}$ preserves the prefix order between strings. That is, If $w' \sqsubseteq w$, then $g_{|w'|}(w') \sqsubseteq g_{|w|}(w)$. Therefore, we define g as the limit of the mappings g_k as $k \to \infty$.

▶ **Definition 27.** *For every $X \in \Sigma^\infty$ let $g(X) = \cap_{n=1}^\infty C_{g(X\upharpoonright n)}$.*

Notice that for any $X \in \Sigma^\infty$ and $k \in \mathbb{N}$, $g_k(X \upharpoonright k) \sqsubseteq g(X)$. For convenience, we abuse the notation and use g to denote the mapping between infinite strings defined above or a mapping between finite strings defined as follows: $g(w) = g_{|w|}(w)$ for any $w \in \Sigma^*$. The intended meaning is clear from the context.

The mapping g induces a polynomial-time samplable distribution ν over Σ^∞. The sampling algorithm M for ν on input 1^n maps a random seed of length $s_0 \cdot 2^{\lceil \log_2(n) \rceil}$ using g to a string of length $2^{\lceil \log_2(n) \rceil}$, and thereafter trims the output to n bits. The number of random bits used by M on input 1^n is at most $2s_0 \cdot n$. Intuitively, because the sampler uses at most $2s_0 n$ random bits on input 1^n, those random bits, together with a fixed $O(1)$-bit description of the sampler, provide a short polynomial-time description of the output prefix. Thus, for ν-a.e. X, we obtain $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(X) \leq 2s_0 < s$. The formal complexity bound is given by the following lemma and corollary.

▶ **Lemma 28.** *For $s_0 = 2^{-m}$ for some $m > 1$, let $G_n : \Sigma^{s_0 n} \to \Sigma^n$ be a PRG that runs in time $t_G(n)$. Let $g : \Sigma^\infty \to \Sigma^\infty$ be the function constructed from G_n as in Definition 26. There exists a constant $c \in \mathbb{N}$ such that for all $X \in \Sigma^\infty$ and for all $n \in \mathbb{N}$, $K_{t_g}(g(X) \upharpoonright 2^n) \leq s_0 \cdot 2^n + c$, where $t_g(n) = t_G(n) \cdot n$.*

Proof. Consider the algorithm M which implements the mapping in Definition 26 as follows.

Algorithm 1 Algorithm M.

Input: A string w.
if $w = \lambda$ **then**
    Output λ
else
    Output 0^{2^m}
end if
for $i = m$ to $\lfloor \log(|w|2^m) \rfloor - 1$ **do**
    Output $G_{2^i}(w[2^{-m} \cdot 2^i, 2^{-m} \cdot 2^{i+1} - 1])$
end for

Since G_n runs in $t_G(n)$ time, it follows that M runs in $t_g(n) = t_G(n) \cdot n$ time. Also, for any $X \in \Sigma^\infty$ and $n \geq m$, M outputs the string $g(X) \upharpoonright 2^n$ on input $X \upharpoonright 2^{-m} \cdot 2^n = X \upharpoonright s_0 \cdot 2^n$. Therefore, it follows that $K_{t_g}(g(X) \upharpoonright 2^n) \leq s_0 \cdot 2^n + c$ where the constant c depends only on the length of any representation of M in the prefix language of the fixed universal Turing machine. ◀

We obtain the following corollary.

▶ **Corollary 29.** *For $s_0 = 2^{-m}$ for some $m > 1$, let $G_n : \Sigma^{s_0 n} \to \Sigma^n$ be a PRG that runs in time $t_G(n)$. Let $g : \Sigma^\infty \to \Sigma^\infty$ be the function constructed from G_n as in Definition 26. There exists a constant $c \in \mathbb{N}$ such that for all $X \in \Sigma^\infty$ and for all $i \in \mathbb{N}$, $K_{t_g}(g(X) \upharpoonright i) \leq 2s_0 \cdot i + 2\log(i) + c'$, where $t_g(n) = t_G(n)n$.*

Proof. From Lemma 28, $K_{t_g}(g(X) \upharpoonright 2^{\lceil log(i) \rceil}) \leq s_0 \cdot 2^{\lceil log(i) \rceil} + c'$. With $K_n(i)$ extra bits, we specify the index i at which $g(X) \upharpoonright 2^{\lceil log(i) \rceil}$ needs to be trimmed. Thus, we obtain a program that outputs $g(X) \upharpoonright i$ in $t_g(n)$-time. There is a trivial prefix-free program which given $2\log(i)$ bits of information, runs in linear time and prints i. Hence, the value of $K_n(i) \leq 2\log(i) + O(1)$. Since $2^{\lceil log(i) \rceil} \leq 2i$, there exists $c' > 0$ such that $K_{t_g}(g(X) \upharpoonright i) \leq 2s_0 \cdot i + K_n(i) + c \leq 2s_0 \cdot i + 2\log(i) + c'$. ◀

4.1.2 From betting gains to a finite-length distinguisher

Suppose toward a contradiction that there exist $s' \in (s, 1/2)$ and a polynomial-time ν-approximable s'-supergale d' with $\nu(S^\infty(d')) > 0$. Since $2s_0 < s < s'$, this is in particular a gale exponent above the compression rate obtained from the PRG-induced source. Using standard gale manipulations, we convert d' into a polynomial-time ν-approximable supermartingale $\tilde{d}$ that exhibits large multiplicative gains between dyadic checkpoints 2^{n-1} and 2^n infinitely often along every sequence in some positive-ν-measure set. This conversion is carried out in two stages, established in the following lemmas.

▶ **Lemma 30.** *For $s' \in [0, \infty)$ if $d' : \Sigma^* \to \mathbb{R}_{\geq 0}$ is a $t(n)$-time computable ν-approximable s'-supergale, then for any s'' satisfying $s' \leq s'' \leq 1$ and $2^{s''} \in \mathbb{Q}$, there exists an exact $t(n)poly(n)$-time ν-approximable supermartingale $\tilde{d} : \Sigma^* \to \mathbb{R}_{\geq 0}$ such that for any $X \in S^\infty(d')$,*

$$\limsup_{n \to \infty} 2^{-(1-s'')n} \tilde{d}(X \upharpoonright n) = \infty.$$

Proof. Let $s'' \geq s'$ such that $2^{s''} \in \mathbb{Q}$. We know that d' is a $t(n)$-time ν-approximable s''-supergale that succeeds on X (see Observation 4.4 from [27]). Let $\bar{d}$ be defined as $\bar{d}(w) = d'(w)2^{(1-s'')|w|}$. Note that as d' is ν-approximable, it follows that $\bar{d}$ is ν-approximable.

From Observation 3.2 in [28], it follows that $\bar{d}$ is a supermartingale such that,

$$\limsup_{n\to\infty} \frac{\bar{d}(X \upharpoonright n)}{2^{(1-s'')n}} = \infty.$$

Since $2^{s''}$ is rational, so is $2^{(1-s'')}$. Hence, $\bar{d}$ is an $t(n)poly(n)$-time ν-approximable supermartingale. ◀

▶ **Lemma 31.** *For $s' < 1/2$, let $d' : \Sigma^* \to \mathbb{R}_{\geq 0}$ be an $t(n)$-time ν-approximable s'-supergale. Let $\tilde{s}$ be such that $\tilde{s} > s'$, $2^{\tilde{s}} \in \mathbb{Q}$. Then, there exists an $t(n)poly(n)$-time ν-approximable supermartingale $\tilde{d} : \Sigma^* \to \mathbb{R}_{\geq 0}$ such that for any $X \in S^\infty(d')$, there exists infinitely many $n \in \mathbb{N}$ such that $\tilde{d}(X \upharpoonright 2^n) > 2^{(1-\tilde{s})2^{n-1}} \tilde{d}(X \upharpoonright 2^{n-1})$.*

Proof. Choose $s'' \in (s', 1/2)$ such that $s' < 2s' < 2s'' < \tilde{s}$ and $2^{s''} \in \mathbb{Q}$. Let $\tilde{d}$ be the $t(n)poly(n)$-time ν-approximable supermartingale from Lemma 30 such that

$$\limsup_{n\to\infty} \frac{\tilde{d}(X \upharpoonright n)}{2^{(1-s'')n}} = \infty.$$

Suppose, toward a contradiction, that for some $m > 0$ and every $n > m$,

$$\tilde{d}(X \upharpoonright 2^n) \leq 2^{(1-\tilde{s})2^{n-1}} \tilde{d}(X \upharpoonright 2^{n-1}).$$

Then, for any $n > m$ and $\ell \leq 2^n$,

$$\tilde{d}(X \upharpoonright (2^n + \ell)) \leq 2^\ell 2^{(1-\tilde{s})\sum_{i=m}^{n-1} 2^i} \tilde{d}(X \upharpoonright 2^m) = 2^\ell 2^{(1-\tilde{s})(2^n - 2^m)} \tilde{d}(X \upharpoonright 2^m).$$

If also $\tilde{d}(X \upharpoonright (2^n + \ell)) \geq 2^{(2^n+\ell)(1-s'')}$, then rearranging gives

$$2^{\ell s''} \geq \frac{2^{(1-\tilde{s})2^m}}{\tilde{d}(X \upharpoonright 2^m)} 2^{2^n(\tilde{s}-s'')}.$$

Since $\ell \leq 2^n$, we obtain

$$2^{2^n(\tilde{s}-2s'')} \leq \frac{\tilde{d}(X \upharpoonright 2^m)}{2^{(1-\tilde{s})2^m}}. \tag{1}$$

If there are infinitely many k such that $\tilde{d}(X \upharpoonright k) \geq 2^{(1-s'')k}$, then (1) holds for infinitely many n. But this is impossible for all large enough n, since $\tilde{s} > 2s''$. Therefore, for infinitely many n,

$$\tilde{d}(X \upharpoonright 2^n) > 2^{(1-\tilde{s})2^{n-1}} \tilde{d}(X \upharpoonright 2^{n-1}),$$

as required. ◀

Let M be a polynomial-time ν-approximator for $\tilde{d}$. We then define a polynomial-time algorithm A that, on input length 2^n, samples a random r, forms the corresponding prefix $w' = g(r) \upharpoonright 2^n$, and checks via M whether appending the input causes $\tilde{d}$ to jump by the required multiplicative factor.

4.1.3 The distinguisher algorithm

Let $\tilde{d}$ be the $t(n) \cdot poly(n)$-time ν-approximable martingale obtained from Lemma 31. Let M be the Turing machine that ν-approximates d. That is for all $n \in \mathbb{N}$, $\nu_n\{w \in \Sigma^n : M(w) \notin [\mathbf{c} \cdot d(w), d(w)]\} \leq n^{-k}$ where $\mathbf{c} \in \mathbb{Q}, k \in \mathbb{N}$ are constants such that $\mathbf{c} \leq 1$ and $k \geq 1$. We use M to build a distinguisher algorithm A that breaks the PRG $\{G_n\}$.

Algorithm 2 Algorithm A.

Input: A string w where $|w| = 2^n$
$r \leftarrow \text{random}(s_0 \cdot 2^n)$ ▷ Generate a random seed of length $s_0 \cdot 2^n$
for $i = m$ to $n-1$ **do**
    $v_i \leftarrow G_{2^i}(r[s_0 \cdot 2^i, s_0 \cdot 2^{i+1} - 1])$ ▷ Use r and G_{2^i} to get a PRG output of length 2^i
end for
$w' \leftarrow 0^{2^m} v_1 \dots v_n$ ▷ Concatenate all v_i
return 1 **if** $M(w'w) \geq \mathbf{c} \cdot 2^{(1-\tilde{s})|w|} \cdot M(w')$.

4.1.4 Performance on PRG outputs

A remaining subtlety is that the martingale guarantee is only *infinitely often* along each sequence in a positive-measure set. We upgrade this to a *noticeable fraction at infinitely many lengths* using a Borel–Cantelli argument, which yields infinitely many n for which a polynomially-large set of seeds r witnesses the required dyadic-block gain. At these lengths, the ν-approximation property of M ensures that A outputs 1 with inverse-polynomial probability on PRG outputs.

We first define the function f such that $f(X, n) = 1$ if $\tilde{d}$ has a sufficiently large gain in the n-th block of X, and $f(X, n) = 0$ otherwise.

▶ **Definition 32.** *Let $f : \Sigma^\infty \times \mathbb{N} \to \{0,1\}$ be defined by specifying $f(X,n) = 1$ if $\tilde{d}(X \upharpoonright 2^n) > 2^{(1-\tilde{s})2^n} \tilde{d}(X \upharpoonright 2^{n-1})$ and 0 otherwise.*

The following fact is a consequence of the Borel Cantelli Lemma [2].

▶ **Lemma 33.** *Let $f : \Sigma^\infty \times \mathbb{N} \to \{0,1\}$ and $\mathcal{S} \subseteq \Sigma^\infty$ such that $\nu(\mathcal{S}) > 0$. If for every $X \in \mathcal{S}$, there exist infinitely many $n \in \mathbb{N}$ such that $f(X,n) = 1$, then for any $c > 1$, there exist infinitely many $n \in \mathbb{N}$, such that $\nu(\{X : f(X,n) = 1\}) > n^{-c}$.*

Now, we apply the above lemma to the martingale $\tilde{d}$ from Lemma 30 to obtain the following. Recall that ν was defined at the start of this section as the distribution obtained by applying g to random seeds.

▶ **Lemma 34.** *Given $\tilde{d} : \Sigma^* \to \mathbb{R}_{\geq 0}$, if for all sequences $X \in \mathcal{S}$ where $\nu(\mathcal{S}) > 0$, there exists infinitely many $n \in \mathbb{N}$ such that $\tilde{d}(X \upharpoonright 2^n) > 2^{(1-\tilde{s})2^{n-1}} \tilde{d}(X \upharpoonright 2^{n-1})$, then it holds that for infinitely many n,*

$$\Pr_{x \sim \Sigma^{s_0 \cdot 2^n}}[\tilde{d}(g(x)) > 2^{(1-\tilde{s})2^n} \tilde{d}(g(x \upharpoonright s_0 \cdot 2^{n-1}))] \geq \frac{1}{n^2}. \tag{2}$$

This follows directly from Lemma 33. Indeed, by the hypothesis and the definition of f, every $X \in \mathcal{S}$ satisfies $f(X,n) = 1$ for infinitely many n. Since $\nu(\mathcal{S}) > 0$, Lemma 33 implies that this event has ν-measure at least n^{-c} for infinitely many n, for every $c > 1$. Unwinding the definition of f and using that ν is induced by applying g to uniformly random seeds gives the displayed probability bound; taking $c = 2$ completes the proof.

Now we analyze the performance of the algorithm on outputs of the PRG. Let $n+1$ be one of the lengths at which the condition in Lemma 34 holds. Let $w', w \in \Sigma^{2^n}$ such that $w = G_{2^n}(x)$ for some $x \in \Sigma^{s_0 2^n}$ and $w' = g(r)$ for some $r \in \Sigma^{s_0 2^n}$.

In order for the condition $M(w'w) \geq \mathbf{c} \cdot 2^{(1-\tilde{s})|w|} \cdot M(w')$ to hold, it suffices that $\tilde{d}(w'w) \geq 2^{(1-\tilde{s})|w|} \cdot \tilde{d}(w')$ and $M(w'w) \geq \mathbf{c} \cdot \tilde{d}(w'w)$ and $M(w) \leq \tilde{d}(w)$.

From Lemma 34, it follows that the first condition holds with probability at least $(n+1)^{-2}$ among x, r selected uniformly at random from $\Sigma^{s_0 \cdot 2^n} \times \Sigma^{s_0 \cdot 2^n}$ and taking $w = g(x), w' = g(r)$. Since $\tilde{d}$ is ν-approximable, for some $k \in \mathbb{N}$, on input $w \in \Sigma^{2^n}$, the computations of all $M(w)$ in Algorithm 2 outputs a value in $[\mathbf{c} \cdot \tilde{d}(w), \tilde{d}(w)]$ with probability at least $1 - 2^{-(kn-1)}$ according to the distribution ν_{2^n}. Note that the distribution according to ν_{2^n} corresponds to g applied to the uniform distribution on $2^{s_0 n}$.

Therefore it follows that for infinitely many n,

$$\Pr_{(x,r)\sim\Sigma^{s_0\cdot 2^n}\times\Sigma^{s_0\cdot 2^n}}[A(G_{2^n}(x), r) = 1] \geq \frac{1}{(n+1)^2} - \frac{1}{2^{kn}}.$$

Hence, taking $N = 2^n$, for infinitely many n,

$$\Pr_{x\sim\Sigma^{s_0\cdot N}}[A(G_N(x)) = 1] \geq \frac{1}{(\log N + 1)^2} - \frac{1}{N^k}. \tag{3}$$

4.1.5 Performance on uniformly random inputs

Finally, we show that A outputs 1 with exponentially small probability on a uniformly random input of the same length. This uses the Kolmogorov inequality for supermartingales to bound how many extensions can force a multiplicative jump of the required size, together with the fact that M fails to approximate $\tilde{d}$ only on a ν-small set.

▶ **Lemma 35** (Kolmogorov Inequality [27]). *Let $d : \Sigma^* \to \mathbb{R}_{\geq 0}$ be a supermartingale. For any $w' \in \Sigma^*$ and $n, c \in \mathbb{N}$, the number of strings $w \in \Sigma^n$ such that $d(w'y) \geq c \cdot d(w')$ for some $y \sqsubseteq w$ is less than or equal to $2^n/c$.*

We need an upper bound on the number of strings $w \in \Sigma^{2^n}$ and $r \in \Sigma^{s_0 2^n}$ such that $A(w, r) = 1$. This happens when $M(w'w) \geq \mathbf{c} \cdot 2^{(1-\tilde{s})|w|} \cdot M(w')$.

For any $n \in \mathbb{N}$, take any $r \in \Sigma^{s_0 \cdot 2^n}$, and let $w' = g(r)$. Using the Kolmogorov Inequality for Martingales (Lemma 35), the number of strings $w \in \Sigma^{2^n}$ such that $\tilde{d}(w'w) \geq \mathbf{c}^2 \cdot 2^{|w|(1-\tilde{s})}\tilde{d}(w')$ is less than $2^{|w|}/(\mathbf{c}^2 \cdot 2^{|w|(1-\tilde{s})})$.

For the remaining set of strings, we have that $\tilde{d}(w'w) < \mathbf{c}^2 \cdot 2^{|w|(1-\tilde{s})}\tilde{d}(w')$. If we have that $M(w'w) \leq \tilde{d}(w'w)$ and $M(w') \geq \mathbf{c} \cdot \tilde{d}(w')$, it follows that $A(w, r) = 0$.

We know that, $\nu\{w \in \Sigma^{2^n} : M(w) \notin [\mathbf{c} \cdot \tilde{d}(w), \tilde{d}(w)]\} \leq 2^{-nk}$ for some $k \in \mathbb{N}$. From this, it follows that $\Pr_{r\in\Sigma^{s_0\cdot N}}[M(w') < \mathbf{c} \cdot \tilde{d}(w')] \leq 2^{-nk}$.

Notice that $\{w \in \Sigma^j : M(w) \notin [\mathbf{c} \cdot \tilde{d}(w), \tilde{d}(w)]\} \subseteq \mathrm{supp}(\nu_j)$ for every j. From the block-wise definition of the mapping g, for any fixed $w' \in g(\Sigma^{2^n})$, we obtain that there are at most $2^{s_0 \cdot 2^n}$ strings in the set $\{w'w\}_{w\in\Sigma^{2^n}}$ contained in $\mathrm{supp}(\nu_{2^{n+1}})$. Therefore, for any fixed w', the number of strings $w \in \Sigma^{2^n}$ such that $M(w'w) \notin [\mathbf{c} \cdot \tilde{d}(w'w), \tilde{d}(w'w)]$ is at most $2^{s_0 \cdot 2^n}$. Therefore, we have that

$$\Pr_{x\sim\Sigma^N}\Pr_{r\in\Sigma^{s_0\cdot N}}[A(x, r) = 1] < \mathbf{c}^{-2} \cdot 2^{-N(1-\tilde{s})} + N^{-k} + 2^{-(1-s_0)N}.$$

Therefore, it follows that for any $n \in \mathbb{N}$,

$$\Pr_{x\sim\Sigma^N}[A(x) = 1] < \mathbf{c}^{-c} \cdot 2^{-N(1-\tilde{s})} + N^{-k} + 2^{-(1-s_0)N}.$$

Comparing the two cases yields that for infinitely many lengths, A distinguishes PRG outputs from uniform with non-negligible advantage, contradicting pseudorandomness. Hence $\nu(S^\infty(d')) = 0$ for every polynomial-time ν-approximable s'-supergale with $s' \in (s, 1/2)$.

4.1.6 Proof of Lemma 25

Proof. If one-way functions exist then there exists PRG $G_n : \Sigma^{s_0 n} \to \Sigma^n$ running in time $t_G(n) \in \mathrm{poly}(n)$ [Theorem 18]. Using $\{G_n\}$, we construct a polynomial-time samplable distribution ν over Σ^∞ as above. By Corollary 29 and the choice of s_0, we have $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(X) \leq 2s_0 < s$ for almost every $X \sim \nu$.

From the arguments in Sections 4.1.4 and 4.1.5, we obtain the following consequence. Suppose there exists some $s' \in (s, 1/2)$ and a polynomial-time ν-approximable s'-supergale d such that $\nu(S^\infty(d)) > 0$. Then there exists a distinguisher algorithm A running in time $t_A(n) \in \mathrm{poly}(n)$ such that, for infinitely many $N \in \mathbb{N}$ and for all $k \in \mathbb{N}$,

$$\Pr_{x \in \Sigma^{s_0 \cdot N}}[A(G_N(x)) = 1] - \Pr_{x \in \Sigma^N}[A(x) = 1] > \frac{1}{(\log N + 1)^2} - \frac{2}{N^k} - \frac{1}{\mathbf{c}^2 \cdot 2^{N(1-\tilde{s})}} - \frac{1}{2^{N(1-s_0)}}.$$

Let c be any constant such that $t_A(n) \leq n^c$ and for all large enough N,

$$\frac{1}{(\log N + 1)^2} - \frac{2}{N^k} - \frac{1}{\mathbf{c}^2 \cdot 2^{N(1-\tilde{s})}} - \frac{1}{2^{N(1-s_0)}} \geq \frac{1}{N^c}.$$

It follows that G_N is not a $(N^c, 1/N^c)$ PRG for infinitely many N. Therefore the family $\{G_n\}_{n \in \mathbb{N}}$ is not a PRG, contradicting the assumption that one-way functions exist. This completes the proof of the lemma. ◀

4.2 Infinitely-often one-way functions from dimension gaps

We now explain the reverse direction: an almost-everywhere dimension gap over an efficiently samplable source can be converted into infinitely-often one-way functions.

▶ **Lemma 36.** *If there exist an $s < 1$ and a polynomial-time samplable distribution ν over Σ^∞ such that:*

1. *$\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(X) \leq s$ for ν-almost every X.*
2. *There exists $\hat{s} \in (s, 1]$ such that for every $s' \in (s, \hat{s})$ and every polynomial-time ν-approximable s'-supergale d, we have $\nu(S^\infty(d)) = 0$.*

Then infinitely-often one-way functions exist.

Proof. Assume the hypothesis, and toward a contradiction assume that infinitely-often one-way functions do not exist. Fix reals $s < s' < s'' < s''' < \hat{s}$ with $2^{s'''} \in \mathbb{Q}$, and define

$$d(w) := 2^{s'''|w|}\, \nu(w).$$

It is straightforward that d is an s'''-supergale. The low-$\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}$ condition implies that with ν-probability close to 1, draws $X \sim \nu$ have many prefixes $X \upharpoonright n$ with small time-bounded description length (say $\leq s'n$ for all large n). For each n, there are at most $2^{s'n}$ such "typical" strings of length n. If all these typical strings had ν_n-mass $< 2^{-s''n}$ for all large n, then the total ν_n-mass of typical length-n strings would be at most $2^{s'n} \cdot 2^{-s''n} = 2^{-(s''-s')n}$, contradicting that typical strings carry almost all the mass. Therefore, with *positive* ν-probability, infinitely many n satisfy $\nu_n(X \upharpoonright n) \geq 2^{-s''n}$, and along those n we have $d(X \upharpoonright n) \geq 2^{(s'''-s'')n}$; hence $\nu(S^\infty(d)) > 0$.

We now formalize the positive-measure conclusion used above.

▶ **Lemma 37.** $\nu\{X \in \Sigma^\infty : \exists^\infty n \text{ such that } \nu_n(X \upharpoonright n) \geq 2^{-n \cdot s''}\} > 0$.

Notice that if $\nu_n(w) \geq 2^{-n \cdot s''}$ then $d(w) \geq 2^{n(s'''-s'')}$. Therefore from Lemma 37 we obtain that $\nu\{X \in \Sigma^\infty : \limsup_{n\to\infty} d(X \upharpoonright n) = \infty\} > 0$.

On the other hand, under the assumption that infinitely-often one-way functions do not exist, we invoke a distributional estimation theorem for polynomial-time samplable distributions. Distributional estimation results of this kind go back to Impagliazzo and Luby [20] and Impagliazzo and Levin [19]. Ilango, Ren, and Santhanam [18] give an algorithm that, for every polynomial-time samplable distribution ensemble $\mathcal{D} = \{\mathcal{D}_n\}$, estimates $\mathcal{D}_n(w)$ within a constant multiplicative factor on all but a $\mathcal{D}_n$-small exceptional set of samples $w \sim \mathcal{D}_n$. Nanashima and Hirahara [13] prove a substantially stronger form, giving a multiplicative $(1 \pm \delta)$ approximation (for arbitrarily small $\delta > 0$) on almost all samples from $\mathcal{D}_n$. Such distributional estimation results can be used to obtain efficient algorithms for universal extrapolation and Solomonoff's inductive inference. For our purposes in this paper, the constant-factor guarantee from [18] is sufficient, and we apply it to the particular samplable distribution $\nu_n(w) = \nu(C_w)$. Combining this estimator with a standard gale normalization step yields a polynomial-time ν-approximator for $\nu(C_w)$ on $\mathrm{supp}(\nu_n)$, and therefore for $d(w) = 2^{s'''|w|}\nu(C_w)$ (using $2^{s'''} \in \mathbb{Q}$). Thus d is a polynomial-time ν-approximable s'''-supergale with $\nu(S^\infty(d)) > 0$, contradicting item (2). Hence infinitely-often one-way functions must exist. The formal estimation result and the approximability argument are given next.

▶ **Theorem 38** ([18, 20, 19, 13])**.** *Assume that no infinitely-often one-way functions exist. Let $\mathcal{D} = \{\mathcal{D}_n\}$ be a polynomial-time samplable distribution and $q \geq 1$ be an arbitrary constant. Then, there exists a probabilistic polynomial-time algorithm $\mathcal{A}$ and constant $c < 1$ such that for all n,*

$$\Pr_{x\sim\mathcal{D}_n}[c \cdot \mathcal{D}_n(x) \leq \mathcal{A}(x) \leq \mathcal{D}_n(x)] \geq 1 - O\left(\frac{1}{n^q}\right).$$

▷ Claim 39. d is polynomial-time ν-approximable.

Proof. Let S be the machine that samples the distribution ν such that for every n, $\Pr_r[S(1^n, r) = w] = \nu_n(w)$ for every $w \in \Sigma^n$. Let c' be the constant such that S uses $n^{c'}$ random bits to sample a string of length n.

Assume that no infinitely-often one-way functions exist. Then, the function $f : \Sigma^* \to \Sigma^*$ defined as

$$f(x) = S(1^{|x|^{1/c'}}, x)$$

is not an infinitely-often weak one-way function. Hence for any $q \geq 1$ there exist an algorithm $\mathcal{I}$ such that for all n,

$$\Pr_{r\sim\Sigma^{n^{c'}}}[f(\mathcal{I}(f(r))) = f(r)] = \Pr_{w\sim\nu_n}[f(\mathcal{I}(w)) = w] \geq 1 - O\left(n^{-q}\right). \tag{4}$$

From our assumption that infinitely-often one-way functions do not exist and Theorem 38, we obtain that for any $q \geq 1$, there exists a probabilistic polynomial-time algorithm $\mathcal{A}$ and constant $c < 1$ such that for all n,

$$\Pr_{w\sim\nu_n}[c \cdot \nu_n(w) \leq \mathcal{A}(w) \leq \nu_n(w)] \geq 1 - O\left(n^{-q}\right). \tag{5}$$

Finally, we give the algorithm $\mathcal{M}$ that ν-approximates d. $\mathcal{M}$ on input $w \in \Sigma^n$ runs $\mathcal{I}$ on w. If $f(\mathcal{I}(w)) \neq w$, $\mathcal{M}$ outputs 0. Else, $\mathcal{M}$ runs $\mathcal{A}$ on w and outputs $2^{s'''|w|}\mathcal{A}(w)$. Since $2^{s'''} \in \mathbb{Q}$, $\mathcal{M}$ runs in polynomial time. For any w such that $\nu(w) = 0$ (i.e. $w \notin \mathrm{supp}(\nu_n)$), $\mathcal{I}$

will never succeed in finding a pre-image r such that $f(r) = w$ and hence $\mathcal{M}$ always outputs 0. Therefore for every n, $\{w : \mathcal{M}(w) \notin [c \cdot d(w), d(w)]\} \subseteq \text{supp}(\nu_n)$. Now, from (4) and (5) we obtain that,

$$\Pr_{x\sim\nu_n}[c \cdot d(x) \leq \mathcal{M}(x) \leq d(x)] \geq 1 - O\left(\frac{1}{n^q}\right) - O\left(\frac{1}{n^q}\right) \geq 1 - O\left(\frac{1}{n^q}\right).$$

Therefore for every n, $\nu\{w : \mathcal{M}(w) \notin [c \cdot d(w), d(w)]\} \leq O\left(n^{-q}\right)$. Since $\mathcal{A}$ runs in randomized polynomial time and q is arbitrary, $\mathcal{M}$ witnesses the fact that d is a polynomial-time ν-approximable s'''-gale. ◁

The claim shows that d is a polynomial-time ν-approximable s'''-supergale. Since $s''' < \hat{s}$ and $\nu(S^\infty(d)) > 0$, this contradicts the second item in the hypothesis. Therefore, infinitely-often one-way functions exist. ◀

4.3 Dimension gaps over sequences from one-way functions

Lemma 25 is a measure statement: under one-way functions it gives a samplable distribution ν on Σ^∞ for which typical sequences have low $\mathcal{K}^{\text{str}}_{\text{poly}}$, but every polynomial-time ν-approximable gale of exponent $< 1/2$ succeeds only on a ν-null set. We now indicate how this measure-theoretic gap yields an *individual* sequence with $\mathcal{K}^{\text{str}}_{\text{poly}}$ small but $\dim_{\text{P}}$ bounded away from it, and then how the same strategy can be pushed to obtain gaps arbitrarily close to 1.

▶ **Theorem 22.** *If for all $X \in \Sigma^\infty$, $\dim_{\text{P}}(X) = \mathcal{K}_{\text{poly}}(X)$, then no one-way functions exist.*

We first record the gale-combination construction needed to pass from pointwise polynomial-time gales, whose running-time exponents may vary with the sequence, to a single polynomial-time gale on a positive-measure set.

▶ **Lemma 40.** *For any time $t(n) \in \text{poly}(n)$, and $s > 0$ with $2^s \in \mathbb{Q}$, there exists an exact $t(n) \cdot n \cdot \log(n)$-time computable s-gale $\tilde{d}$ such that for all exact $t(n)$-time computable s-gales d, there exist a constant c such that for all $X \in \Sigma^\infty$ and $n \in \mathbb{N}$, $\tilde{d}(X \upharpoonright n) \geq c \cdot d(X \upharpoonright n)$.*

We now prove a stronger theorem. Theorem 22 follows from Theorem 41, since for all sequences $X \in \Sigma^\infty$, $\mathcal{K}_{\text{poly}}(X) \leq \mathcal{K}^{\text{str}}_{\text{poly}}(X)$.

▶ **Theorem 41.** *If one-way functions exist, then for every $s < 1/2$, there exist $X \in \Sigma^\infty$ such that $\mathcal{K}^{\text{str}}_{\text{poly}}(X) \leq s$ and $\dim_{\text{P}}(X) \geq 1/2$.*

Proof. Assume that one-way functions exist. From the proof of Lemma 25, we obtain that for every $s < 1/2$ there exist a polynomial-time samplable distribution ν over Σ^∞ and polynomial t such that:

1. The number of random bits used by the sampler for ν on input 1^n is at most sn.
2. For every $s' \in (s, 1/2)$ and any polynomial-time ν-approximable s'-supergale d, $\nu(S^\infty(d)) = 0$.

Consider the canonical closed full-measure support set $\mathcal{F} = \bigcap_{n\geq 1} \bigcup_{w\in\text{supp}(\nu_n)} C_w$. Now, $\cup_{w\in\text{supp}(\nu_n)} C_w$ is a finite union of cylinder sets in Σ^∞ and hence is a closed set. Since ν is a polynomial-time samplable probability distribution on Σ^∞, $\cup_{w\in\text{supp}(\nu_n)} C_w$ is a superset of $\cup_{w\in\text{supp}(\nu_{n+1})} C_w$ for every n. Since $\mathcal{F}$ is an intersection of a decreasing sequence of non-empty closed sets in the compact space Σ^∞, $\mathcal{F}$ is non-empty (see [35]). Since $\nu(\cup_{w\in\text{supp}(\nu_n)} C_w) = 1$ for every n, using the continuity of probability measures from above [2], we get $\nu(\mathcal{F}) = 1$. Since the number of random bits used by the sampler for ν on input 1^n is at most sn, $\mathcal{K}^{\text{str}}_{\text{poly}}(X) \leq s$ for every $X \in \mathcal{F}$.

For the sake of contradiction, assume that there exists $s' \in (s, 1/2)$ such that $\dim_{\mathrm{P}}(X) < s'$ for every $X \in \mathcal{F}$. So, for every $X \in \mathcal{F}$ there exists an exact computable s'-gale d'_X that runs in time $t_{d'_X}(n) \in \mathrm{poly}(n)$, such that d'_X succeeds on X. We group $\mathcal{F}$ according to a polynomial running-time exponent. Define

$$R_i = \left\{ X \in \mathcal{F} \;\middle|\; \text{there exists an } n^i\text{-time } s'\text{-gale } d \text{ such that } X \in S^\infty(d) \right\},$$

and let $S_i = R_i \setminus R_{i-1}$. Note that $\{S_i\}_{i\in\mathbb{N}}$ is a partition of $\mathcal{F}$ into countably many disjoint subsets. As $\sum_{i=1}^{\infty} \nu(S_i) = 1$, there exists an $k \in \mathbb{N}$ such that $\sum_{i=1}^{k} \nu(S_i) > 0$. Define $\mathcal{S} = \bigcup_{i=1}^{k} S_i$. On this positive-measure subset, all witnessing gales run within a common polynomial time bound. Using Lemma 40, there exists an exact polynomial-time computable s'-gale d that succeeds on all sequences in $\mathcal{S}$. Since d is polynomial-time computable, it is automatically polynomial-time ν-approximable. Now, since we know that $\nu(S^\infty(d)) \geq \nu(\mathcal{S}) > 0$, we obtain a contradiction to Lemma 25. Hence, it must be the case that there exists a sequence $X \in \mathcal{F}$ with $\dim_{\mathrm{P}}(X) > s'$. Since s' was arbitrary, the theorem follows. ◀

The same argument also gives an almost-sure version over the samplable source obtained from Lemma 25.

▶ **Lemma 42.** *If one-way functions exist then for some $s < 1/2$, there exist a polynomial-time samplable distribution ν over Σ^∞ such that:*

1. *The number of random bits used by the sampler for ν on input 1^n is at most sn.*
2. *For every $s' \in (s, 1/2)$ and polynomial-time ν-approximable s'-supergale d, $\nu(S^\infty(d)) = 0$.*
3. *$\nu\{X \in \Sigma^\infty : \mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(X) \leq s$ and $\dim_{\mathrm{P}}(X) \geq 1/2 > s\} = 1$.*

We now strengthen the conclusion by amplifying the gap. The above route naturally yields the threshold $1/2$ because the forward implication only rules out success of *ν-approximable* gales below $1/2$. To push the separation to $1 - \varepsilon$, we keep the same blueprint (construct a low-$\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}$ source and diagonalize against efficient betting strategies), but we modify the diagonalization so that it is carried out against *exact* computable gales. This avoids the accumulation of approximation slack, which becomes the bottleneck near dimension 1, and allows us to push the lower bound on $\dim_{\mathrm{P}}$ to $1 - \varepsilon$ while keeping $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}$ below ε.

▶ **Theorem 23.** *If one-way functions exist, then for any $\epsilon > 0$, there exists $X \in \Sigma^\infty$ such that $\dim_{\mathrm{P}}(X) - \mathcal{K}_{\mathrm{poly}}(X) \geq 1 - \epsilon$.*

We prove the following stronger statement independently since it does not follow directly using the techniques in the proofs of Lemma 25 and Theorem 41. The main obstacle is that techniques involving approximable gales introduce non-negligible approximation errors while trying to achieve gaps close to 1. In the proof of the below theorem we instead employ computable gales. Whether Lemma 25 can be strengthened to yield such a gap remains an open question.

▶ **Theorem 43.** *If one-way functions exist, then for every $\epsilon > 0$ there exist set $\mathcal{F}$ and a sequence X such that $\dim_{\mathrm{P}}(\mathcal{F}) - \mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(\mathcal{F}) \geq 1 - \epsilon$ and $\dim_{\mathrm{P}}(X) - \mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(X) \geq 1 - \epsilon$ respectively.*

The argument follows the same overall blueprint as the proof of Theorem 22: we use the PRG-induced mapping g to produce sequences with low $\mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}$, and then diagonalize against efficient betting strategies by converting a successful s'-gale into a martingale that must exhibit large multiplicative gains at dyadic checkpoints, which yields a distinguisher for the underlying PRG. The only substantive change is that, to obtain gaps arbitrarily close to

1, we carry out the diagonalization using exact polynomial-time computable gales rather than ν-approximable gales, so that no approximation slack accumulates when we test many intermediate prefixes. A similar argument yields the corresponding separation statement for sets, by applying the same support/compactness reduction and the same diagonalization idea at the level of collections rather than individual sequences.

We obtain the following corollary of Theorem 43. This result provides a strong negative answer to the open question posed by Stull in [36] under the assumption that one-way functions exist.

▶ **Corollary 44.** *If one-way functions exist then for every $\epsilon < 1/2$ there exists $X \in \Sigma^\infty$ such that $\mathcal{K}_{\mathrm{poly}}(X) \leq \mathcal{K}^{\mathrm{str}}_{\mathrm{poly}}(X) < \epsilon < 1 - \epsilon < \dim_{\mathrm{P}}(X) \leq \mathrm{Dim}_{\mathrm{P}}(X)$.*

5 Open Problems and Discussion

Our findings suggest several promising directions for future research. A key open question is whether weaker hardness assumptions, such as P $\neq$ NP, DistNP $\not\subseteq$ AvgP or DistNP $\not\subseteq$ AvgBPP, can be characterized in terms of separations of polynomial-time dimension, as defined using time-bounded s-gales and time-bounded Kolmogorov complexity in appropriate settings. Related work on time-bounded Kolmogorov complexity has also revealed connections with average-case hardness and one-way functions [25, 9]. Specifically, can these hardness assumptions be equivalently expressed through if-and-only-if statements involving polynomial-time dimension?

A different direction is to seek finite-length versions of these phenomena. Polynomial-time dimension is asymptotic, while the pseudorandomness assumptions used in this paper are finite-length indistinguishability assumptions. It would be interesting to define finite-length measures based on efficient online betting strategies, or equivalently log-loss prediction on n-bit samples, and compare them with finite-time Kolmogorov complexity rates. Can one obtain finite-length separation theorems between efficient betting and efficient compression that characterize one-way functions, average-case hardness, or other standard assumptions?

Another natural direction concerns extraction and dimension amplification. Standard randomness extractors transform weak finite sources into distributions close to uniform [37], while dimension measures asymptotic randomness rates along individual infinite sequences. Can seeded extractors, condensers, or related pseudorandomness objects be lifted to transformations that increase polynomial-time dimension for typical outputs of efficiently samplable infinite sources? A complementary question is whether cryptographic hardness can be used to construct efficiently samplable sources whose finite prefixes have substantial entropy, but whose polynomial-time dimension cannot be increased by these transformations.

Our results reveal a duality between the non-robustness of polynomial-time dimension and the existence of one-way functions. An interesting open question is whether notions of polynomial-time randomness, defined via betting algorithms and Kolmogorov incompressibility of prefixes (see [36] for more details), capture equivalent concepts of randomness. Polynomial-time betting algorithms that succeed on specific sequences do not necessarily provide guarantees on the rate of capital accumulation in general. Consequently, investigating the robustness of polynomial-time randomness and its potential connections to cryptographic and complexity-theoretic primitives may require fundamentally new approaches.

References

1 Krishna B Athreya, John M Hitchcock, Jack H Lutz, and Elvira Mayordomo. Effective strong dimension in algorithmic information and computational complexity. *SIAM journal on computing*, 37(3):671–705, 2007. doi:10.1137/S0097539703446912.

2 Patrick Billingsley. *Probability and measure*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, Inc., New York, third edition, 1995. A Wiley-Interscience Publication.

3 Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Found. Trends Theor. Comput. Sci.*, 2(1):1–106, December 2006. doi:10.1561/0400000004.

4 Chris Bourke, John M. Hitchcock, and N. V. Vinodchandran. Entropy rates and finite-state dimension. *Theoretical Computer Science*, 349(3):392–406, 2005. doi:10.1016/J.TCS.2005.09.040.

5 Jack J. Dai, James I. Lathrop, Jack H. Lutz, and Elvira Mayordomo. Finite-state dimension. *Theoretical Computer Science*, 310(1-3):1–33, 2004. doi:10.1016/S0304-3975(03)00244-5.

6 W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976. doi:10.1109/TIT.1976.1055638.

7 Rodney G. Downey and Denis R. Hirschfeldt. *Algorithmic randomness and complexity*. Theory and Applications of Computability. Springer, New York, 2010. doi:10.1007/978-0-387-68441-3.

8 Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 416–426. ACM, 1990. doi:10.1145/100216.100272.

9 Halley Goldberg and Valentine Kabanets. Consequences of Randomized Reductions from SAT to Time-Bounded Kolmogorov Complexity. In Amit Kumar and Noga Ron-Zewi, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2024)*, volume 317 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 51:1–51:19, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX/RANDOM.2024.51.

10 Oded Goldreich. *Foundations of Cryptography*. Cambridge University Press, 2001.

11 Oded Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008. doi:10.1017/CBO9780511804106.

12 Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984. doi:10.1016/0022-0000(84)90070-9.

13 Shuichi Hirahara and Mikito Nanashima. Learning in pessiland via inductive inference. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 447–457. IEEE, 2023. doi:10.1109/FOCS57990.2023.00033.

14 J.M. Hitchcock and N.V. Vinodchandran. Dimension, entropy rates, and compression. In *Proceedings. 19th IEEE Annual Conference on Computational Complexity, 2004.*, pages 174–183, 2004. doi:10.1109/CCC.2004.1313835.

15 John M. Hitchcock. Effective fractal dimension: Foundations and applications. Ph.D. thesis, Iowa State University, 2003.

16 John M. Hitchcock and N.V. Vinodchandran. Dimension, entropy rates, and compression. *Journal of Computer and System Sciences*, 72(4):760–782, 2006. doi:10.1016/j.jcss.2005.10.002.

17 Johan HÅstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999. doi:10.1137/S0097539793244708.

18 Rahul Ilango, Hanlin Ren, and Rahul Santhanam. Robustness of average-case meta-complexity via pseudorandomness. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1575–1583, 2022. doi:10.1145/3519935.3520051.

19 R. Impagliazzo and L. A. Levin. No better ways to generate hard np instances than picking uniformly at random. In *Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science*, pages 812–821, 1990. doi:10.1109/FSCS.1990.89604.

20 R. Impagliazzo and M. Luby. One-way functions are essential for complexity based cryptography. In *30th Annual Symposium on Foundations of Computer Science*, pages 230–235, 1989. doi:10.1109/SFCS.1989.63483.

21 Valentine Kabanets and Jin-Yi Cai. Circuit minimization problem. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, STOC '00, pages 73–79, New York, NY, USA, 2000. Association for Computing Machinery. doi:10.1145/335305.335314.

22 Andrei N Kolmogorov. Three approaches to the quantitative definition ofinformation'. *Problems of information transmission*, 1(1):1–7, 1965.

23 L. A. Levin. The tale of one-way functions. *Probl. Inf. Transm.*, 39(1):92–103, January 2003. doi:10.1023/A:1023634616182.

24 Ming Li and Paul Vitányi. *An introduction to Kolmogorov complexity and its applications*, volume 3. Springer, 2008.

25 Yanyi Liu and Rafael Pass. On one-way functions and kolmogorov complexity. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1243–1254, 2020. doi:10.1109/FOCS46700.2020.00118.

26 María López-Valdés and Elvira Mayordomo. Dimension is compression. In Joanna Jędrzejowicz and Andrzej Szepietowski, editors, *Mathematical Foundations of Computer Science 2005*, volume 3618 of *Lecture Notes in Computer Science*, pages 676–685, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. doi:10.1007/11549345_58.

27 Jack H. Lutz. Dimension in complexity classes. *SIAM J. Comput.*, 32(5):1236–1259, 2003. doi:10.1137/S0097539701417723.

28 Jack H. Lutz. The dimensions of individual strings and sequences. *Information and Computation*, 187(1):49–79, 2003. doi:10.1016/S0890-5401(03)00187-1.

29 Elvira Mayordomo. A Kolmogorov complexity characterization of constructive Hausdorff dimension. *Inform. Process. Lett.*, 84(1):1–3, 2002. doi:10.1016/S0020-0190(02)00343-5.

30 Moni Naor. Bit commitment using pseudorandomness. *J. Cryptol.*, 4(2):151–158, January 1991. doi:10.1007/BF00196774.

31 Andre Nies. *Computability and Randomness*. Oxford University Press, Inc., USA, 2009.

32 K.W. Regan and D. Sivakumar. Probabilistic martingales and bptime classes. In *Proceedings. Thirteenth Annual IEEE Conference on Computational Complexity (Formerly: Structure in Complexity Theory Conference) (Cat. No.98CB36247)*, pages 186–200, 1998. doi:10.1109/CCC.1998.694604.

33 John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Symposium on the Theory of Computing*, 1990. URL: https://api.semanticscholar.org/CorpusID:10858937.

34 Halsey Lawrence Royden and Patrick Fitzpatrick. *Real analysis*, volume 32. Macmillan New York, 1988.

35 George F. Simmons. *Introduction to Topology and Modern Analysis*. McGraw-Hill, New York, 1963.

36 Donald M. Stull. Resource bounded randomness and its applications. In Johanna N. Y. Franklin and Christopher P. Porter, editors, *Algorithmic Randomness: Progress and Prospects*, Lecture Notes in Logic, pages 301–348. Cambridge University Press, Cambridge, 2020.

37 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/0400000010.

38 Andrew C. Yao. Protocols for secure computations. In *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, pages 160–164, 1982. doi:10.1109/SFCS.1982.38.

Recovering Planted Colorings in Sublinear Time

Weronika Wrzos-Kaminska
EPFL, Lausanne, Switzerland

Abstract

We study the problem of recovering a planted k-coloring in sublinear time. Given an expander G with a planted coloring, the goal is to efficiently construct a small-space data structure that allows consistent color queries: given a vertex v, the algorithm quickly returns the color of v according to the planted solution.

We work in the adversarial planted coloring model of David and Feige [STOC 2016], where an adversary chooses a d-regular spectral λ-expander G on n vertices and plants a balanced k-coloring by partitioning the vertices into k equal parts and deleting all edges within each part. This model generalizes the earlier random graph models studied by Blum and Spencer [J. Algorithms 1995] and Alon and Kahale [STOC 1994].

We give the first sublinear-time algorithm for recovering planted colorings in this model. In the adjacency list model, our algorithm has preprocessing time and space $\widetilde{O}\left(n^{1/2+O(1/\log(d/\lambda))}\right)$, and produces a data structure that answers color queries in time $\widetilde{O}\left(n^{1/2+O(1/\log(d/\lambda))}\right)$. With a high constant probability, the resulting labeling agrees with the planted coloring on all but an $O(\sqrt{\lambda/d})$ fraction of vertices, up to a permutation of the k colors.

Our algorithm gives sublinear time inner product access to the bottom eigenspace of the normalized adjacency matrix by using random walks, which allows us to adapt the classical spectral approach of Alon and Kahale in sublinear time.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases sublinear algorithms, spectral algorithms, graph coloring

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.45

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2608.02338`

1 Introduction

Graph coloring is a fundamental problem in theoretical computer science. In the k-coloring problem, the goal is to assign each vertex a color from $[k]$ so that adjacent vertices receive different colors. Finding k-colorings in graphs for $k \geq 3$ is among the first computational problems shown to be NP-hard [24, 18]. This has motivated extensive work on approximation algorithms and on identifying assumptions under which the problem can be solved efficiently [33, 5, 7, 6, 8, 1, 23, 27, 28, 2, 13, 3, 12, 25, 29, 26, 9, 20].

One influential line of work studies planted coloring models, where a hidden coloring is embedded into a graph, and the task is to recover (approximately) the planted solution. Blum and Spencer [6] and Alon and Kahale [1] considered random graph models with a planted k-coloring, and showed that polynomial-time spectral algorithms can recover the planted k-coloring with high probability. David and Feige [12] extended this to a more challenging adversarial setting, in which an adversary chooses an expander and plants a balanced k-coloring by deleting all edges within each color class. They showed that, even in this adversarial setting, the spectral approach of Alon and Kahale recovers the planted coloring on a 0.99 fraction of the vertices, while exact recovery is NP-hard.

While planted coloring models have been extensively studied in the polynomial-time regime, the question of *sublinear-time* recovery remains largely unexplored. This motivates the central question of this work:


© Weronika Wrzos-Kaminska;
licensed under Creative Commons License CC-BY 4.0

Is it possible to recover a planted k-coloring in sublinear time?

Of course, any algorithm that explicitly outputs a coloring requires $\Omega(n)$ time simply to write down the labels. We therefore focus on constructing a *coloring oracle*: a small-space randomized data structure that quickly answers queries of the form "what color does vertex v have?". The oracle must be *consistent*, meaning that its answers agree with a single underlying assignment of colors (not necessarily a proper k-coloring), and *approximately correct*, meaning that the oracle disagrees with the planted coloring on only a small fraction of vertices, up to a permutation of colors (see Definition 8 for a formal definition of coloring oracle).

We consider a variant of the adversarial planted model of [12]. In this model, an adversary chooses a d-regular spectral λ-expander (i.e., eigenvalues of its adjacency matrix, other than the top eigenvalue d, have absolute value at most λ). The adversary also chooses a balanced partitioning of the vertices into k color classes, for a constant k. The planted instance is obtained by deleting all the monochromatic edges, i.e., all edges that have both endpoints in the same color class. Additionally, we require that the partitioning is chosen so that the degrees in the planted instance are balanced, in the sense that all vertices have degree $\Omega(d)$ after deleting the monochromatic edges (see Definition 6 for a formal definition of the input model).

A d-regular graph has $\lambda_1 = d$ and that the second largest eigenvalue in absolute value satisfies $\lambda = \Omega(\sqrt{d})$ [30]. Random d-regular graphs are essentially best possible spectral expanders, satisfying $\lambda = O(\sqrt{d})$ with high probability [14, 17, 16]. The adversarial planted coloring model therefore generalizes random planted models, in which the host graph is drawn at random and/or the planted partitioning is chosen uniformly at random.

Our contribution

We give the first sublinear-time algorithm for this planted coloring problem. Our result works in the adversarial planted model, and therefore also applies to the corresponding random planted models. Specifically, in the adversarial planted k-coloring model, we construct a coloring oracle with $\widetilde{O}(n^{1/2+O(1/\log(d/\lambda)}$ preprocessing time and space, where n is the number of vertices. The oracle answers each color query in $\widetilde{O}(n^{1/2+O(1/\log(d/\lambda))})$ time and, with a high constant probability, agrees with the planted coloring on at least a 0.99 fraction of the vertices.

▶ **Theorem 1.** *Let $k \geq 3$ be a constant, let c_k be a sufficiently small constant (depending on k), and let C_k be a sufficiently large constant (depending on k). For every $C_k < d < n$ and every $\lambda \leq c_k d$, there exists a randomized algorithm for the adversarial planted k-coloring model (Definition 6) that constructs a coloring oracle using $\widetilde{O}\big(n^{1/2+O(1/\log(d/\lambda))}\big)$ preprocessing time and space, where n denotes the number of vertices in the graph. For every query vertex v, the oracle outputs a color in $[k]$ in time $\widetilde{O}\big(n^{1/2+O(1/\log(d/\lambda))}\big)$, and its answers are consistent across queries. With probability at least 0.9, the oracle agrees with the planted coloring (up to a permutation of the colors) on all but at most $O\left(\sqrt{\frac{\lambda}{d}}n\right)$ vertices.*

In particular, the running time is $\approx n^{1/2+o(1)}$ when $\lambda/d = o(1)$. For dense graphs with $d = n^\delta$ and $\lambda = O(\sqrt{d})$ for a constant δ, the running time becomes $\widetilde{O}(n^{1/2})$. As in prior works on planted coloring, we treat the number of colors k as a constant.

Our algorithm may be viewed as a sublinear-time implementation of the classical spectral algorithm of [1]. The key technical ingredient is providing sublinear-time dot-product access to the bottom eigenspace of the adjacency matrix, which we achieve by adapting recent techniques from spectral clustering [22].

Related work on planted coloring

Blum and Spencer [6] and Alon and Kahale [1] studied the problem of planted k-coloring in random graphs, and showed that exact recovery is possible in polynomial time. [12] studied the role of randomness in the planted model, and showed that exact recovery is possible if the coloring is planted at random in an adversarial spectral expander, and that approximate recovery is possible even when the coloring is adversarial. [29] proved that approximate recovery is possible in low threshold rank graphs for pseudorandom colorings. Recently, [4, 9, 20] further considered the setting in which the coloring is planted in a one-sided expander or a low threshold rank graph.

Related work on sublinear spectral algorithms

Our techniques are closely related to work on sublinear spectral clustering [11, 10, 19, 32, 15, 22]. Most relevant to our work is [22], which uses a low-degree polynomial of the random walk matrix to provide access to the top eigenspace of the adjacency matrix under a clusterability assumption. Although this setting differs from ours (our graphs are not clusterable, and the signal lies in a different part of the spectrum), we adopt their general approach. [31] and [21] applied spectral techniques to approximate max-cut (and more generally max-2Lin) in sublinear time on expanders and clusterable graphs. The work of [31] also showed that testing whether an expander is 3-colorable versus far from 3-colorable requires $\Omega(n)$ time. We note that the hard instances in [31] are constant degree expanders with expansion $\phi = \Omega(1/d)$, whereas our results require stronger spectral expansion.

2 Technical Overview

Our algorithm can be viewed as a sublinear implementation of the spectral approaches of Alon and Kahale [1] and David and Feige [12]. These works show that planted colorings can be approximately recovered from the bottom eigenspace of the adjacency matrix. The main difficulty is that computing this exactly requires reading the whole graph and computing the eigenvectors.

Recent progress on sublinear spectral algorithms has focused on approximating inner products of spectral embeddings associated with the *top* eigenspace of a graph, typically under strong clusterability assumptions [19, 22]. Our main technical contribution is to extend this line of work to the setting of graph coloring by providing sublinear-time access to the *bottom* eigenspace of the adjacency matrix. While bottom eigenvectors have long played a central role in polynomial-time spectral algorithms, sublinear-time methods for accessing this part of the spectrum have remained largely undeveloped.

The spectral approach of [1] and [12]

We start by reviewing the classical approach of [1] and [12]. These works showed that an approximate coloring can be recovered from the bottom eigenspace of the adjacency matrix of the planted input graph. Specifically, the bottom $k-1$ eigenvalues of the normalized

adjacency matrix have value $-\frac{1}{k-1} \pm O(\sqrt{\lambda/d})$, while all remaining eigenvalues (except for the trivial top eigenvalue 1) are bounded in absolute value by $O(\sqrt{\lambda/d})$ [1]. The coloring can be approximately recovered from the spectral embedding in this bottom eigenspace.

▶ **Definition 2** (Spectral embedding). *Let $\bar{A} = D^{-1/2}AD^{-1/2}$ denote the normalized adjacency matrix of the input graph G', and let $U_{[k-1]} \in \mathbb{R}^{n\times(k-1)}$ denote the matrix whose columns are the bottom $k-1$ eigenvectors of $\bar{A}$. For every vertex $v \in V$, the spectral embedding of v, denoted $f_v \in \mathbb{R}^{k-1}$, is the v-th row of $U_{[k-1]}$, i.e.,*

$$f_v \coloneqq U_{[k-1]}^\top \mathbb{1}_v.$$

▶ Remark 3. The spectral embeddings f_v are not uniquely defined. However, the dot products $\langle f_u, f_v\rangle$, and hence also the Euclidean distances $\|f_u - f_v\|_2$, are uniquely defined (see Remark 17).

As shown in [12], spectral embeddings of vertices belonging to the same color class are close to each other, while spectral embeddings of vertices belonging to different color classes are far from each other. Specifically, for all but an $O(\lambda/d)$ fraction of vertices, the following holds[2]:

$$\|f_u - f_v\|_2^2 = \begin{cases} \leq \dfrac{k}{70n}, & \text{if } \chi(u) = \chi(v), \\ \geq \dfrac{k}{2n}, & \text{if } \chi(u) \neq \chi(v). \end{cases} \tag{1}$$

By Equation (1), approximate recovery of the planted coloring can be achieved using the following two steps:

1. Identify a set of *color representatives* $c_1 \in C_1, \ldots, c_k \in C_k$, i.e., one vertex from each color class.
2. Given a query vertex u, output the color of the closest color representative

$$\textsc{Color}(u) = \arg\min_{i\in[k]} \|f_u - f_{c_i}\|_2^2.$$

The main challenge is to perform this in sublinear time, without explicitly computing the full spectral embedding.

Spectral embeddings via polynomials

In order to approximate inner products $\langle f_u, f_v\rangle$ between spectral embeddings (and hence also the Euclidean distances), we will approximate the relevant projection matrix by a low-degree polynomial in the random walk matrix. The inner products can then be approximated by collision counting. This idea was used in [22] in the context of sublinear-time spectral clustering, and we adapt their general approach. Let $\bar{A} = U\Sigma U^\top$ denote the eigendecomposition of the normalized adjacency matrix $\bar{A}$. We are interested in approximating

$$\langle f_u, f_v\rangle = \mathbb{1}_u^\top U_{[k-1]} U_{[k-1]}^\top \mathbb{1}_v,$$

[1] [12] analyze the unnormalized adjacency matrix. Since our algorithm is based on random walks, it is more convenient for us to work with the normalized adjacency matrix. An analogous statement for the normalized adjacency matrix $\bar{A}$ follows by a similar argument (see Lemma 16).

[2] Equation (1) can be extracted from the analysis of [12] for embeddings of the *unnormalized* adjacency matrix, although it is not stated there explicitly in this form. We prove an analogous guarantee for the normalized adjacency matrix (see Lemma 20).

where $U_{[k-1]}U_{[k-1]}^\top$ is the projection onto the eigenspace corresponding to eigenvalues with value $\approx -1/(k-1)$. To this end, we seek a polynomial $p(x) = \sum_{t\geq 0} c_t x^t$ of the form $p(x) = x^t q(x)$, where $t = \Omega\left(\frac{\log n}{\log(d/\lambda)}\right)$, such that the coefficients c_t are small and the following properties hold:

- $p(x) \approx 1$ when $\left|x + \frac{1}{k-1}\right| \leq O\left(\sqrt{\frac{\lambda}{d}}\right)$, and
- $p(x) \approx 0$ when $|x| \leq O\left(\sqrt{\frac{\lambda}{d}}\right)$ or $x = 1$.

With such a polynomial, the matrix $p(\bar{A}) = Up(\Sigma)U^\top$ acts as an approximate projector onto the bottom $k-1$ eigenvectors. As a result,

$$\begin{aligned}\langle f_u, f_v\rangle &= \mathbb{1}_u^\top U_{[k-1]}U_{[k-1]}^\top \mathbb{1}_v \\ &\approx \mathbb{1}_u^\top U p(\Sigma)^2 U^\top \mathbb{1}_v = \langle p(\bar{A})\mathbb{1}_u, p(\bar{A})\mathbb{1}_v\rangle = \left\langle \sum_t c_t \bar{A}^t \mathbb{1}_u, \sum_t c_t \bar{A}^t \mathbb{1}_v \right\rangle \\ &= \sum_{t,t'} c_t c_{t'} \langle \bar{A}^t \mathbb{1}_u, \bar{A}^{t'} \mathbb{1}_v\rangle.\end{aligned}$$

Each inner product $\langle \bar{A}^t \mathbb{1}_u, \bar{A}^{t'} \mathbb{1}_v\rangle$ can be estimated efficiently using random walks. Indeed, let $M = AD^{-1}$ denote the random walk transition matrix. For $t = \Omega\left(\frac{\log n}{\log(d/\lambda)}\right)$, inner products of the form $\langle M^t \mathbb{1}_u, M^{t'} \mathbb{1}_v\rangle$ can be approximated using $\approx \sqrt{n}$ random walks via standard collision-counting arguments. Using the relation $\bar{A} = D^{-1/2}MD^{1/2}$, inner products involving powers of $\bar{A}$ can be approximated similarly.

It is crucial that the coefficients c_t of the polynomial p are small: each term $\langle \bar{A}^t \mathbb{1}_u, \bar{A}^{t'} \mathbb{1}_v\rangle$ must be estimated to precision inverse polynomial in $|c_t c_{t'}|$, and hence the overall running time depends polynomially on the size of coefficients.

[22] used a similar idea to cluster graphs. In their setting, the input graph is assumed to be clusterable, meaning that the top k eigenvalues lie in the interval $[1-\epsilon, 1]$ for some error parameter ϵ, while all remaining eigenvalues are bounded away from 1. The goal is to approximate embeddings in this *top* eigenspace of the graph, which they achieve by constructing a polynomial that approximates the indicator on $[1-\epsilon, 1]$.

A key challenge with adapting their construction to our setting is that the coefficients of their polynomial grows superpolynomially in their error parameter ϵ, and can be as large as $\approx (1/\epsilon)^{O(\log(1/\epsilon))} n^{O(\epsilon)}$. Since the running time depends polynomially on the coefficient size, the algorithm is sublinear only for relatively large values of ϵ.

In our setting, the relevant analogue of the error parameter is $\sqrt{\lambda/d}$, and a superpolynomial dependence on $\sqrt{d/\lambda}$ would lead to too large running times. For example, for random d-regular graphs with $d = n^{\Omega(1)}$, this would lead to running times that are superpolynomial in n.

To overcome this issue, we modify the polynomial construction of [22] and give a more careful analysis of the resulting polynomial. This gives a polynomial that approximates an indicator function around $-1/(k-1)$, while having coefficients with absolute at value most $n^{O(1/\log(d/\lambda))}$. In particular, the coefficient size, and hence the running time, decreases as λ/d decreases.

▶ **Theorem 4.** *Let $\delta, C > 0$ be constants. For every $\frac{10C\log n}{\log(d/\lambda)} \leq t \leq O\left(\frac{\log n}{\log(d/\lambda)}\right)$, there exists a polynomial p of the form $p(x) = x^t q(x)$, where q is a polynomial of degree $O(\sqrt{\lambda/d}\log n)$, such that the coefficients of p are bounded in absolute value by $n^{O\left(\frac{1}{\log(d/\lambda)}\right)}$, and*

1. *$|p(x) - 1| \leq \delta$ for $x \in \left[-\frac{1}{k-1} - C\sqrt{\frac{\lambda}{d}}, -\frac{1}{k-1} + C\sqrt{\frac{\lambda}{d}}\right]$*
2. *$|p(x)| \leq n^{-2}$ for $|x| \leq C\sqrt{\frac{\lambda}{d}}$.*

In fact, our proof establishes a more general result: for any constant $\alpha \in [-1, 1]$ with $\alpha \neq 0$ and any $0 < \epsilon \ll |\alpha|$, we construct a polynomial approximating an indicator on $[\alpha - \epsilon, \alpha + \epsilon]$ with similar coefficient bounds (see Theorem 28). The proof of Theorem 4 is presented in Section 5.4.

While this polynomial successfully maps the eigenvalues near $-1/(k-1)$ to values close to 1 and eigenvalues of size at most $\sqrt{\lambda/d}$ to values close to 0, it does not control the value of $p(1)$, which corresponds to the top eigenvalue. We handle this issue by explicitly removing the contribution of the top eigenvector. Since the top eigenvector of $\bar{A}$ is $e_1 = D^{1/2}\mathbb{1}/\sqrt{2m}$, its contribution can be computed exactly, and we can therefore apply the polynomial to the matrix $\bar{A} - e_1 e_1^\top$, in which the top eigenvalue has been shifted to zero.

The algorithms for approximating $\langle f_u, f_v \rangle$ (Algorithm 2) and $\|f_u - f_v\|_2^2$ (Algorithm 3) are presented in Section 4. Their guarantees (Lemma 10 and Lemma 11) are proved in Section 5.2.

Finding color representatives

Given access to pairwise distances between spectral embeddings, it remains to pick one representative vertex from each planted color class. The polynomial-time approach of [12] finds such representatives by repeatedly sampling k random vertices until exactly one comes from each class. However, their method for verifying whether a given sample is valid requires $\Omega(n)$ time, which is too costly in our setting. We instead sample $\Theta(k \log k)$ vertices and build a small "similarity graph" on the sample, connecting two sampled vertices when their embeddings are similar. This is similar to approaches used for clustering [11, 32].

The similarity graph decomposes into exactly k connected components (one per color class) with a high constant probability, so choosing one vertex from each component yields the required representatives. We present the algorithm for obtaining color representatives in Section 4.1.

3 Preliminaries and Notation

We start by formally defining the graph access model and the input model.

Graph access model. We work in the adjacency-list query model. In this model, an algorithm can perform neighbor queries, i.e., for the i-th neighbor of any vertex, and degree queries, i.e., for the degree of any vertex. We additionally assume that the number of edges m is given as part of the input to the algorithm.

▶ **Definition 5** (λ-expander). *Let $\lambda_1(A_G) \geq \lambda_2(A_G) \geq \cdots \geq \lambda_n(A_G)$ denote the eigenvalues of the unnormalized adjacency matrix A_G of a graph G. Say that G is a λ-expander if $\max\{|\lambda_2(A_G)|, |\lambda_n(A_G)|\} \leq \lambda$.*

▶ **Definition 6** (Adversarial planted k-coloring). *Let $k \geq 3$ be a constant. An adversary chooses an n-vertex d-regular λ-spectral expander $G = (V, E)$ together with a balanced partition $\mathcal{C} = (C_1, \ldots, C_k)$ of V, such that $|C_i| = n/k$ for all $i \in [k]$ and such that every vertex has at least $\Omega(d)$ neighbors outside its color class. The partition $\mathcal{C}$ defines a* planted k-coloring $\chi : V \to [k]$ *by setting $\chi(v) = i$ if $v \in C_i$.*

The planted instance is obtained by deleting all monochromatic edges from G. That is, let

$$E' \coloneqq \{(u, v) \in E : \chi(u) \neq \chi(v)\},$$

and let $G' = (V, E')$ denote the resulting k-colorable graph. We write $G' = P_{\mathcal{C}}(G)$ when G' is obtained this way from the graph G and the partition $\mathcal{C}$.

▶ Remark 7 (Model assumptions). The model above is almost identical to the planted adversarial model of [12], except that we impose the additional condition that every vertex has at least a small constant fraction of its neighbors outside of its own color class. This ensures that after deleting monochromatic edges, every vertex in G' has degree $\Omega(d)$. Since the typical degree in G' is $(1-\frac{1}{k})d$, this is a mild restriction that only rules out vertices that are almost isolated by the adversary. In particular, if the color classes are selected *at random* and $d = \Omega(\log n)$, then this holds with high probability. The purpose of this assumption is to ensure that the normalized adjacency matrix of G' has the required spectral properties. Our sublinear-time algorithm is based on random walks and therefore relies on spectral properties of the normalized adjacency matrix. In contrast, prior polynomial-time spectral algorithms for planted k-coloring work directly with the *unnormalized* adjacency matrix, and therefore do not need this additional balanced degree condition.

For simplicity of presentation, Definition 6 assumes that the planted coloring is exactly balanced, i.e., $|C_i| = n/k$ for all $i \in [k]$. Our results extend to approximately balanced planted colorings satisfying $|C_i|/|C_j| = O(1)$ for all $i, j \in [k]$. The exact balance assumption is consistent with the standard formulation in prior work (e.g., [6, 1, 12]).

Similar to prior work, we assume that $k = O(1)$, and suppress the dependency on k in the O-notation.

The goal is to construct a coloring oracle, as defined below.

▶ **Definition 8** (Coloring oracle). *A coloring oracle is a randomized data structure $\mathcal{O}$ that, given query access to a graph G' from the adversarial planted k-coloring model, provides consistent query access to a function $\textsc{Color} : V \to [k]$.*

We say that $\mathcal{O}$ defines an ε-approximate coloring if, with probability at least 0.9 over the random bits of $\mathcal{O}$, there exists a permutation $\pi : [k] \to [k]$ such that

$$|\{v \in V : \textsc{Color}(v) \neq \pi(\chi(v))\}| \leq \varepsilon n,$$

where χ is the planted coloring.

Notation

Given the input graph $G' = (V, E')$, we let m denote the number of edges $|E'|$. We write A for the unnormalized adjacency matrix of G' and D for the diagonal degree matrix. Write $\bar{A} = D^{-1/2}AD^{-1/2}$ for its normalized adjacency matrix, and write $M = AD^{-1}$ for the random walk matrix. Observe that $M = D^{1/2}\bar{A}D^{-1/2}$.

Let $\lambda_1(\bar{A}) \geq \lambda_2(\bar{A}) \geq \cdots \geq \lambda_n(\bar{A})$ denote the eigenvalues of $\bar{A}$, and let $e_1(\bar{A}), \ldots, e_n(\bar{A})$ denote a corresponding orthonormal basis of eigenvectors. When it is clear that we are talking about the matrix $\bar{A}$, we omit the dependence on $\bar{A}$ from the notation.

For a vertex $v \in V$, we write $\mathbb{1}_v \in \mathbb{R}^n$ for the indicator of v, i.e., the vector with entry 1 at index v and 0 everywhere else. Similarly, for a set $S \subseteq V$, we write $\mathbb{1}_S \in \mathbb{R}^n$ for the indicator vector of S. We use the notation $\mathbb{1} \in \mathbb{R}^n$ for the vector with all entries equal to 1.

▶ **Observation 9.** *$\bar{A}D^{1/2}\mathbb{1} = D^{1/2}\mathbb{1}$. In particular,*

$$e_1(\bar{A}) = \frac{D^{1/2}\mathbb{1}}{\|D^{1/2}\mathbb{1}\|_2} = \frac{D^{1/2}\mathbb{1}}{\sqrt{2m}} \qquad \text{and} \qquad \lambda_1(\bar{A}) = 1.$$

For a polynomial q, write $\deg(q)$ to denote the degree of q and $\max|\text{coeff}(q)|$ to denote the absolute value of the largest (in absolute value) coefficient of q.

We use $\widetilde{O}$ notation to suppress $\operatorname{polylog} n$ factors.

4 The Algorithm

In this section, we present the algorithm. The algorithm SpectralDotProduct (Algorithm 2) approximates inner products $\langle f_u, f_v \rangle$ of spectral embeddings. It runs $\approx \sqrt{n}$ random walks of various lengths starting from u and v, and then combines the resulting collision counts with the coefficients of the polynomial–p from Theorem 4.

To change from the random walk matrix $M = AD^{-1}$ to the normalized adjacency matrix $\bar{A} = D^{-1/2}MD^{1/2}$, we appropriately reweigh collisions according to the matrix $D^{-1/2}$. Finally, we subtract the term $\frac{1}{2m}$ in order to cancel out the contribution of the top eigenvalue.

Algorithm 1 RandomWalks(r, t, u).

Run r random walks of length t starting from u
Let $\hat{p}_u(v)$ be the fraction of random walks that end at v ▷ vector $\hat{p}_u$ has support at most r
return $\hat{p}_u$

Algorithm 2 SpectralDotProduct(u, v).

$t_{\min} \leftarrow O\left(\frac{\log n}{\log(d/\lambda)}\right)$ ▷ shortest random walk length
$t_\Delta \leftarrow O(\sqrt{\lambda/d}\log n)$ ▷ number of different walk lengths
$r \leftarrow \widetilde{O}\left(n^{1/2+O\left(\frac{1}{\log(d/\lambda)}\right)}\right)$ ▷ number of random walks
for $t = t_{\min}, \dots t_{\min} + t_\Delta$ **do**
    $\hat{p}_u^t \leftarrow$ RandomWalks(r, t, u) ▷ vector $\hat{p}_u^t$ has support at most r
    $\hat{p}_v^t \leftarrow$ RandomWalks(r, t, v) ▷ vector $\hat{p}_v^t$ has support at most r
end for
return $\sqrt{\deg(u)}\sqrt{\deg(v)}\sum_{t,t'} c_t c_{t'}\left(\left\langle D^{-1/2}\hat{p}_u^t, D^{-1/2}\hat{p}_v^{t'}\right\rangle - \frac{1}{2m}\right)$
▷ c_t is the coefficient of x^t in the polynomial $p(x)$ from Theorem 4

SpectralDotProduct has the following guarantee: outside a small deterministic set B of "bad" vertices, SpectralDotProduct(u, v) approximates $\langle f_u, f_v \rangle$ up to an additive error of $O(1/n)$. The set B is defined formally in Definition 19 and consists of vertices whose degree or spectral embedding is atypical relative to their planted color class. We bound the size of B in Lemma 20.

▶ **Lemma 10** (Correctness of SpectralDotProduct). *The procedure SpectralDotProduct (Algorithm 2) runs in time* $\widetilde{O}\left(n^{1/2+O\left(\frac{1}{\log(d/\lambda)}\right)}\right)$. *For every* $u, v \in V \setminus B$,

$$\Pr\left[\left|\textsc{SpectralDotProduct}(u, v) - \langle f_u, f_v \rangle\right| \le \frac{1}{80}\frac{k}{n}\right] \ge 0.9,$$

where the probability is over the internal randomness of SpectralDotProduct.

We prove the correctness guarantee of SpectralDotProduct (Lemma 10) in Section 5.2.

To compute distances $\|f_u - f_v\|_2^2$, we use the algorithm SpectralDistance (Algorithm 3). The algorithm boosts the success probability of SpectralDotProduct and approximates $\|f_u - f_v\|_2^2 = \|f_u\|_2^2 + \|f_v\|_2^2 - 2\langle f_u, f_v \rangle$ by approximating each of the terms.

Algorithm 3 SpectralDistance(u, v).

for $i = 1, \dots O(\log n)$ **do**
  $\|f_u\|^2_{\text{apx},i} \leftarrow$ SpectralDotProduct(u, u) ▷ Approximate $\|f_u\|_2^2$
  $\|f_v\|^2_{\text{apx},i} \leftarrow$ SpectralDotProduct(v, v) ▷ Approximate $\|f_v\|_2^2$
  $\langle f_u, f_v\rangle_{\text{apx},i} \leftarrow$ SpectralDotProduct(u, v) ▷ Approximate $\langle f_u, f_v\rangle$
  $\|f_u - f_v\|^2_{\text{apx},i} \leftarrow \|f_u\|^2_{\text{apx},i} + \|f_v\|^2_{\text{apx},i} - 2\langle f_u, f_v\rangle_{\text{apx},i}$
end for
return $\text{median}_i(\|f_u - f_v\|^2_{\text{apx},i})$

▶ **Lemma 11** (Correctness of SpectralDistance)**.** *The procedure SpectralDistance (Algorithm 3) runs in time* $\widetilde{O}\left(n^{1/2+O\left(\frac{1}{\log(d/\lambda)}\right)}\right)$*. With probability at least* $1 - n^{-100}$*,*

$$\left|\textsc{SpectralDistance}(u, v) - \|f_u - f_v\|_2^2\right| \le \frac{k}{20n} \qquad \text{for all } u, v \in V \setminus B,$$

where the probability is over the internal randomness of SpectralDistance.

As a corollary, SpectralDistance(u, v) can be used to determine if u and v belong to the same color class, for all $u, v \in V \setminus B$.

▶ **Corollary 12.** *With probability at least* $1 - n^{-100}$*, the following holds for all* $u, v \in V \setminus B$*:*

- *If* $\chi(u) = \chi(v)$*, then* SpectralDistance$(u, v) \le \frac{k}{4n}$*.*
- *If* $\chi(u) \neq \chi(v)$*, then* SpectralDistance$(u, v) \ge \frac{k}{2n}$*.*

We prove Lemma 11 and Corollary 12 in Section 5.3. In particular, Algorithm 3 can be used as a coloring oracle: given a set of color representatives $c_1, \dots, c_k$, where each c_i belongs to a different color class and none of them lies in the set of bad vertices B, one can answer each query u by assigning the color of its nearest neighbor among the c_i. Algorithm 4 does this.

Algorithm 4 Color$\left(u, \{c_i\}_{i\in[k]}\right)$.
Input: vertex u and set of color representatives $\{c_i\}_{i\in[k]}$ (see Definition 13)
Output: Index $i \in [k]$ such that $\chi(u) = \chi(c_i)$

return $\arg\min_{i\in[k]}$ SpectralDistance(u, c_i)

4.1 Obtaining color representatives

It remains to find a valid set of color representatives. The procedure FindRepresentatives (Algorithm 5) below achieves this. The algorithm samples $\Theta(k \log k)$ vertices uniformly at random, which ensures that, with high probability, we sample at least one vertex from each color class. The algorithm then constructs a "similarity graph" on the sampled vertices, connecting two sampled vertices if their distance is sufficiently small. By the clustering properties of the spectral embeddings (see Lemma 20), with a good probability, vertices from the same color class form a clique in this similarity graph, while vertices from different color classes are disconnected. Finally, the algorithm selects one arbitrary vertex from each connected component of the similarity graph and discards the rest. We show that, with probability at least 0.99, this procedure returns exactly one representative from each color class, yielding a valid set of representatives for use in the coloring oracle. Algorithm 5 is very similar to the preprocessing routines of [32] and [15].

Algorithm 5 FindRepresentatives.

$S \leftarrow$ Multiset of $10k \log k$ vertices sampled uniformly at random from V
$H \leftarrow (S, \emptyset)$ ▷ H will be the same-color graph on S (see definition in full version)
for all $u, v \in S$ **do**
  if SpectralDistance$(u, v) \leq \frac{k}{3n}$ **then**
    Add edge $\{u, v\}$ to H
  end if
end for
$i \leftarrow 1$
while $V(H) \neq \emptyset$ **do**
  Select arbitrary vertex u from $V(H)$ ▷ Select a representative of a color class
  $c_i \leftarrow u$
  $T_i \leftarrow \{u\} \cup N_H(u)$ ▷ The sampled vertices from the same color class as u
  $H \leftarrow H[V(H) \setminus T_i]$ ▷ Remove all vertices from the color class of u
  $i \leftarrow i + 1$
end while
return $c_1, c_2, \ldots$

▶ **Definition 13** (Color Representatives). *A set of vertices $\{c_i\}_{i \in [k]}$ is called a* set of color representatives *if the following conditions hold:*

1. *Every c_i is a good vertex (see Definition 19).*
2. *The vertices have distinct colors, i.e., $\chi(c_i) \neq \chi(c_j)$ for all $i \neq j$.*

▶ **Lemma 14** (Correctness of FindRepresentatives). *The procedure FindRepresentatives (Algorithm 5) runs in time $\widetilde{O}(n^{1/2+O(1/\log(d/\lambda))})$. With probability at least 0.99, it returns a set of k color representatives (as per Definition 13).*

The proof of Lemma 14 follows the analysis of the preprocessing procedure in [15], and is presented in the full version.

5 Analysis

In this section, we analyze the algorithms presented in Section 4 and prove the main result, Theorem 1. The section is organized as follows. In Section 5.1, we define the set B of bad vertices and bound its size. In Section 5.2, we prove Lemma 10, establishing the correctness of SpectralDotProduct. In Section 5.3, we prove Lemma 11 and Corollary 12, establishing the correctness of SpectralDistance. In Section 5.4 we construct the polynomial and prove Theorem 4. Finally, in Section 5.5 we combine everything to prove Theorem 1.

5.1 Properties of the input instance

We start by stating some of the key properties of the input instance G'. The proofs of these properties are included in the full version.

▶ **Observation 15.** *By the model assumption (Definition 6),* $\deg(v) = \Theta(d)$ *for all $v \in G'$ and $\Omega(nd) \leq m \leq nd$.*

▶ **Lemma 16** (Normalized version of Lemma C.4 in [12]). *Let G be a d-regular λ-expander, where $\lambda \le c_k d$ (for some constant c_k), and let $G' = P_{\mathcal{C}}(G)$ (as per Definition 6). Then*

1. *The eigenvalues of the normalized adjacency matrix $\bar{A}$ of G' have the following spectrum:*
   a. $\lambda_1(\bar{A}) = 1$
   b. $\left|\lambda_i(\bar{A}) + \frac{1}{k-1}\right| \le O\left(\sqrt{\frac{\lambda}{d}}\right)$ *for* $n-k+2 \le i \le n$.
   c. $|\lambda_i(\bar{A})| \le O\left(\sqrt{\frac{\lambda}{d}}\right)$ *for* $2 \le i \le n-k+1$
2. *For every $x \in \mathrm{span}(\{\mathbb{1}_{C_i}\}_{i\in[k]})$ with $\langle x, \mathbb{1}\rangle = 0$, there exists $x^* \in \mathrm{span}(\{e_i(G')\}_{i\in[n-k+2,n]})$ such that $\|D^{1/2}x - x^*\|_2^2 = O\left(\sqrt{\frac{\lambda}{d}}\right)\|D^{1/2}x\|_2^2$. Here $C_1, \dots C_k$ denotes the planted partition into color classes.*

[12] proves analogous guarantees to Lemma 16 for the eigenvalues and eigenvectors of the *unnormalized* adjacency matrix (see Lemma C.4 in [12]). Our proof of Lemma 16 is similar and is included in the full version.

▶ Remark 17. It follows from guarantee 1 in Lemma 16 that the space spanned by the bottom $k-1$ eigenvectors of $\bar{A}$ is uniquely defined, i.e., the choice of $U_{[k-1]}$ is unique up to multiplication by an orthogonal matrix $R \in \mathbb{R}^{(k-1)\times(k-1)}$ on the right.

While the choice of f_v for $v \in V$ is not unique, the dot product between the spectral embedding of $u \in V$ and $v \in V$ is well defined, since for every orthogonal $R \in \mathbb{R}^{(k-1)\times(k-1)}$ one has

$$\langle R^\top f_u, R^\top f_v\rangle = (R^\top f_u)^\top (R^\top f_v) = f_u^\top R R^\top f_v = \langle f_u, f_v\rangle.$$

A useful property of the spectral embeddings is that they concentrate around their color class center.

▶ **Definition 18** (Color class center). *For each color class $C_i \in \mathcal{C}$, let μ_i denote the* color class center

$$\mu_i \coloneqq \frac{1}{|C_i|}\sum_{v\in C_i} f_v = \frac{1}{|C_i|} U_{[k-1]}^\top \mathbb{1}_{C_i}.$$

We define a set of *good* vertices consisting of those vertices that are close to their own color class center, far from other color class centers, and whose degree is close to the "typical" value $\frac{d(k-1)}{k}$.

▶ **Definition 19** (Good vertex). *Say that a vertex $v \in C_i$ is* good *if*

1. $\frac{d(k-1)}{k}\left(1 - \sqrt{\frac{\lambda}{d}}\right) \le \deg(v) \le \frac{d(k-1)}{k}\left(1 + \sqrt{\frac{\lambda}{d}}\right)$, *and*
2. $\|f_v - \mu_i\|_2^2 \le \frac{0.05k}{n}$, *and*
3. *for all $j \ne i$, $\|f_v - \mu_j\|_2^2 \ge \frac{k}{n}$.*

Say that the vertex is bad otherwise. Let $B \subseteq V$ denote the set of bad vertices.

We will show that the oracle is correct on all good vertices. Observe that the set B of bad vertices is deterministic once the input graph and planted coloring are fixed. It consists of vertices whose degree or spectral embedding is atypical relative to their planted color class.

▶ **Lemma 20.** *Let B be the set of bad vertices (as per Definition 19). Then*

$$|B| \le O\left(\sqrt{\frac{\lambda}{d}} \cdot n\right).$$

A similar bound for vertices with bad spectral properties was proved in [12]. While their bound is similar, the definition of "bad" vertices in [12] differs slightly from ours. The proof of Lemma 20 is included in the full version.

Additionally, we will use the fact that the ℓ_2 norm of good vertices is not too large.

▶ **Lemma 21.** *If v is good (as per Definition 19), then*

$$\|f_v\|_2^2 = O\left(\frac{1}{n}\right).$$

The proof of Lemma 21 is presented in the full version.

5.2 Analysis of SpectralDotProduct (Proof of Lemma 10)

In this section, we prove the correctness guarantee of SpectralDotProduct, namely Lemma 10. Recall that we wish to apply the polynomial from Theorem 4 to the normalized adjacency matrix $\bar{A}$ in order to approximate the projection $U_{[k-1]}U_{[k-1]}^\top$. However, since Theorem 4 does not provide any guarantees for $p(1)$, we need to shift the top eigenvalue. Therefore, it will be convenient for the analysis to consider the matrix $\bar{A} - e_1 e_1^\top$.

▶ **Definition 22.** *Let*

$$\bar{A}_\star = \bar{A} - e_1 e_1^\top.$$

We start by showing that $\mathbb{1}_u^\top p(\bar{A}_\star)^2 \mathbb{1}_v$ approximates the inner product $\langle f_u, f_v\rangle$. Later, in Claim 27, we will show the output of Algorithm 2 approximates $\mathbb{1}_u^\top p(\bar{A}_\star)^2 \mathbb{1}_v$.

▶ **Lemma 23.** *Let $\delta > 0$ be a sufficiently small constant, let p be the polynomial from Theorem 4 (applied with the parameter δ), and let B be the set of bad vertices from Definition 19. For every $u, v \in V \setminus B$,*

$$\left|\mathbb{1}_u^\top p(\bar{A}_\star)^2 \mathbb{1}_v - \langle f_u, f_v\rangle\right| \le \frac{1}{160}\frac{k}{n}.$$

Proof. The high-level idea is to decompose the matrix $\bar{A}_\star$ according to its spectrum. We separate it into two parts: one corresponding to the bottom $k-1$ eigenvalues, and one corresponding to the remaining $n-k+1$ eigenvalues. By Lemma 16, the bottom $k-1$ eigenvalues are all close to $-1/(k-1)$. Applying the polynomial p to this part therefore "boosts" these eigenvalues up to (approximately) 1, so this part closely approximates the spectral projection $U_{[k-1]}U_{[k-1]}^\top$. On the other hand, the remaining $n-k+1$ eigenvalues have absolute value at most $O(\sqrt{\lambda/d})$ (again by Lemma 16). The polynomial p maps these values close to zero, so the error introduced by this part is small.

We now do this more formally. Let $\bar{A}_\star = U\Sigma^\star U^\top$ denote the eigendecomposition of $\bar{A}_\star$, where $\Sigma^\star = \mathrm{diag}(0, \lambda_2, \lambda_3, \ldots, \lambda_n)$. Observe that $\bar{A}$ and $\bar{A}_\star$ share the same eigenbasis, so the matrix $U_{[k-1]}$ from Definition 2 is the matrix consisting of the last $k-1$ columns of U.

We write $\Sigma^\star_{[k-1]}$ for the $(k-1)\times(k-1)$ diagonal matrix consisting of the bottom $k-1$ diagonal entries of $\Sigma^\star$, and $\Sigma^\star_{\mathrm{top}}$ for the $(n-k+1)\times(n-k+1)$ diagonal matrix consisting of the remaining (top) entries. We let U_{top} denote the matrix formed by the first $n-k+1$ columns of U.

With this notation, we can decompose

$$p(\bar{A}_\star)^2 = Up(\Sigma^\star)^2U^\top = U_{[k-1]}p(\Sigma^\star_{[k-1]})^2U_{[k-1]}^\top \;+\; U_{\mathrm{top}}p(\Sigma^\star_{\mathrm{top}})^2U_{\mathrm{top}}^\top. \tag{2}$$

Recalling from Definition 2 that $\langle f_u, f_v\rangle = \mathbb{1}_u^\top U_{[k-1]} U_{[k-1]}^\top \mathbb{1}_v$, we can bound the error by decomposing $p(\bar{A}_\star)^2$ into the two terms in Equation (2):

$$\begin{aligned}
&\left|\mathbb{1}_u^\top p(\bar{A}_\star)^2 \mathbb{1}_v - \langle f_u, f_v\rangle\right| \\
&= \left|\mathbb{1}_u^\top U p(\Sigma^\star)^2 U^\top \mathbb{1}_v - \mathbb{1}_u^\top U_{[k-1]} U_{[k-1]}^\top \mathbb{1}_v\right| \\
&= \left|\mathbb{1}_u^\top U_{[k-1]} p(\Sigma^\star_{[k-1]})^2 U_{[k-1]}^\top \mathbb{1}_v + \mathbb{1}_u^\top U_{\text{top}} p(\Sigma^\star_{\text{top}})^2 U_{\text{top}}^\top \mathbb{1}_v - \mathbb{1}_u^\top U_{[k-1]} U_{[k-1]}^\top \mathbb{1}_v\right| \\
&= \left|\mathbb{1}_u^\top U_{[k-1]} \left(p(\Sigma^\star_{[k-1]})^2 - I_{k-1}\right) U_{[k-1]}^\top \mathbb{1}_v + \mathbb{1}_u^\top U_{\text{top}} p(\Sigma^\star_{\text{top}})^2 U_{\text{top}}^\top \mathbb{1}_v\right| \\
&\le \left|f_u^\top \left(p(\Sigma^\star_{[k-1]})^2 - I_{k-1}\right)\ f_v\right| + \left|\mathbb{1}_u^\top U_{\text{top}} p\left(\Sigma^\star_{\text{top}}\right)^2 U_{\text{top}}^\top \mathbb{1}_v\right|.
\end{aligned} \tag{3}$$

Here the second equality follows by Equation (2), and the last inequality follows by the triangle inequality and the identity $f_u = U_{[k-1]}^\top \mathbb{1}_u$.

We now bound the first summand in Equation (3). By Lemma 16 and the definition of $\Sigma^\star_{[k-1]}$, the matrix $\Sigma^\star_{[k-1]}$ is a diagonal matrix with entries in $\left[-\frac{1}{k-1} - O\left(\sqrt{\frac{\lambda}{d}}\right), -\frac{1}{k-1} + O\left(\sqrt{\frac{\lambda}{d}}\right)\right]$. Therefore, by Theorem 4, $p(\Sigma^\star_{[k-1]})^2$ is a diagonal matrix with entries in $[1-3\delta, 1+3\delta]$. Hence, $\left\|p(\Sigma^\star_{[k-1]})^2 - I_{k-1}\right\|_2 \le 3\delta$, so

$$\left|f_u^\top (p(\Sigma^\star_{[k-1]})^2 - I_{k-1})\ f_v\right| \le \|f_u\|_2 \left\|\left(p(\Sigma^\star_{[k-1]})^2 - I_{k-1}\right) f_v\right\|_2 \le 3\delta \|f_u\|_2 \|f_v\|_2 \le \frac{1}{320n}, \tag{4}$$

where the last inequality follows by Lemma 21 and the assumption that δ is a sufficiently small constant.

We now bound the second summand in Equation (3). The matrix $U_{\text{top}} p(\Sigma^\star_{\text{top}})^2 U_{\text{top}}^\top$ is positive semidefinite, and therefore

$$\begin{aligned}
\left|\mathbb{1}_u^\top U_{\text{top}} p(\Sigma^\star_{\text{top}})^2 U_{\text{top}}^\top\ \mathbb{1}_v\right| &\le \mathbb{1}_u^\top U_{\text{top}} p(\Sigma^\star_{\text{top}})^2 U_{\text{top}}^\top \mathbb{1}_u + \mathbb{1}_v^\top U_{\text{top}} p(\Sigma^\star_{\text{top}})^2 U_{\text{top}}^\top \mathbb{1}_v \\
&\le \left\|p\left(\Sigma^\star_{\text{top}}\right)^2\right\|_2 \left\|U_{\text{top}}^\top \mathbb{1}_u\right\|_2^2 + \left\|p\left(\Sigma^\star_{\text{top}}\right)^2\right\|_2 \left\|U_{\text{top}}^\top \mathbb{1}_v\right\|_2^2 \\
&\le 2\left\|p\left(\Sigma^\star_{\text{top}}\right)^2\right\|_2,
\end{aligned}$$

where the last inequality uses that $\left\|U_{\text{top}}^\top \mathbb{1}_w\right\|_2^2 \le \|U^\top \mathbb{1}_w\|_2^2 = \|\mathbb{1}_w\|_2^2 = 1$ for all $w \in V$.

By Lemma 16 and the definition of $\Sigma^\star_{\text{top}}$, the matrix $\Sigma^\star_{\text{top}}$ is a diagonal matrix with entries bounded in absolute value by $O(\sqrt{\lambda/d})$ Therefore, by Theorem 4, $\left\|p\left(\Sigma^\star_{\text{top}}\right)^2\right\|_2 \le n^{-2}$. So the second summand in Equation (3) can be bounded as

$$\left|\mathbb{1}_u^\top U_{\text{top}} p(\Sigma^\star_{\text{top}})^2 U_{\text{top}}^\top\ \mathbb{1}_v\right| \le 2\left\|p\left(\Sigma^\star_{\text{top}}\right)^2\right\|_2 \le 2n^{-2} \le \frac{1}{320n}. \tag{5}$$

Combining Equation (3), Equation (4) and Equation (5) gives the claim. ◀

The following lemma shows that the quantity computed by Algorithm 2 has expected value $\mathbb{1}_u^\top p(\bar{A}_\star)^2 \mathbb{1}_v = \sum_{t,t'} c_t c_{t'} \langle \bar{A}_\star^t \mathbb{1}_u, \bar{A}_\star^{t'} \mathbb{1}_v\rangle$.

▶ **Lemma 24.** *For every $u, v \in V$,*

$$\sqrt{\deg(u)}\sqrt{\deg(v)}\left(\langle D^{-1/2} M^t \mathbb{1}_u, D^{-1/2} M^{t'} \mathbb{1}_v\rangle - \frac{1}{2m}\right) = \langle \bar{A}_\star^t \mathbb{1}_u, \bar{A}_\star^{t'} \mathbb{1}_v\rangle.$$

Proof. We start by relating the left-hand side to $\langle \bar{A}^t \mathbb{1}_u, \bar{A}^{t'} \mathbb{1}_v \rangle$. From the definition of $M = AD^{-1}$ and $\bar{A} = D^{-1/2}AD^{-1/2}$, for every t, it holds that $M^t = D^{1/2}\bar{A}^t D^{-1/2}$. Therefore,

$$\begin{aligned}
&\langle D^{-1/2}M^t \mathbb{1}_u, D^{-1/2}M^{t'} \mathbb{1}_v \rangle = \mathbb{1}_u^\top \left(M^t\right)^\top D^{-1} M^{t'} \mathbb{1}_v \\
&= \mathbb{1}_u^\top D^{-1/2} (\bar{A}^t)^\top D^{1/2} D^{-1} D^{1/2} A^{t'} D^{-1/2} \mathbb{1}_v = \left(D^{-1/2}\mathbb{1}_u\right)^\top (\bar{A}^t)^\top A^{t'} D^{-1/2} \mathbb{1}_v \\
&= \frac{\langle \bar{A}^t \mathbb{1}_u, \bar{A}^{t'} \mathbb{1}_v \rangle}{\sqrt{\deg(u)}\sqrt{\deg(v)}}.
\end{aligned} \tag{6}$$

Next, we relate this to $\langle \bar{A}_\star^t \mathbb{1}_u, \bar{A}_\star^{t'} \mathbb{1}_v \rangle$. Recall from Observation 9 that $e_1 = \frac{D^{1/2}\mathbb{1}}{\sqrt{2m}}$. Since $\bar{A}e_1 = e_1$, we have $\bar{A}_\star^t = (\bar{A} - e_1 e_1^\top)^t = \bar{A}^t - e_1 e_1^\top$ for every $t \geq 1$. Therefore,

$$\begin{aligned}
\left\langle \bar{A}_\star^t \mathbb{1}_u, \bar{A}_\star^{t'} \mathbb{1}_v \right\rangle &= \left\langle \bar{A}^t \mathbb{1}_u, \bar{A}^{t'} \mathbb{1}_v \right\rangle - \langle e_1, \mathbb{1}_u \rangle \langle e_1, \mathbb{1}_v \rangle \\
&= \left\langle \bar{A}^t \mathbb{1}_u, \bar{A}^{t'} \mathbb{1}_v \right\rangle - \frac{\langle \mathbb{1}_u, D^{1/2}\mathbb{1} \rangle \langle \mathbb{1}_v, D^{1/2}\mathbb{1} \rangle}{2m} \\
&= \left\langle \bar{A}^t \mathbb{1}_u, \bar{A}^{t'} \mathbb{1}_v \right\rangle - \frac{\sqrt{\deg(u)}\sqrt{\deg(v)}}{2m}.
\end{aligned} \tag{7}$$

Combining Equation (6) and Equation (7) gives the lemma. ◀

To prove Lemma 10, we need to argue that we can estimate the quantity $\sqrt{\deg(u)}\sqrt{\deg(v)} \sum_{t,t'} c_t c_{t'} \left(\left\langle D^{-1/2}M^t \mathbb{1}_u, D^{-1/2}M^{t'} \mathbb{1}_v \right\rangle - \frac{1}{2m} \right)$ to a sufficiently high precision. We will argue this on a term-by-term basis, using the collision counting result stated in Lemma 25.

▶ **Lemma 25** (Weighted Collision Counting)**.** *Let p, q be two distributions $p, q \in \mathbb{R}^n$, let $\xi \in (0,1)$ be the desired precision and let $\zeta \in (0,1)$ be the desired failure probability. Let $w_1, \dots w_n \in (0, w_{\max}]$ and let $W = diag(w_1, \dots w_n)$. One can estimate $p^\top W q$ up to precision $\xi \cdot w_{\max}(\|p\|_2^2 + \|q\|_2^2)$ with success probability $1 - \zeta$ using $r = 5\xi^{-2}\zeta^{-1}(\|p\|_2^2 + \|q\|_2^2)^{-1/2}$ samples from each of p and q. That is, it is possible to compute $\hat{z} \in \mathbb{R}$ such that*

$$\Pr\left[|p^\top W q - \hat{z}| > \xi \cdot w_{\max}\left(\|p\|_2^2 + \|q\|_2^2\right)\right] \leq \zeta.$$

The proof of Lemma 25 is a standard collision counting argument, and is included in the full version.

We will apply Lemma 25 to the distributions $p = M^t \mathbb{1}_u$ and $q = M^{t'} \mathbb{1}_v$, and the weight matrix $W = D^{-1}$. Since both the error and the sample complexity in Lemma 25 depend on $\|p\|_2 = \|M^t \mathbb{1}_u\|_2$, we need to bound the norm $\|M^t \mathbb{1}_u\|_2$ for $u \in V \setminus B$. This is achieved by Lemma 26 below.

▶ **Lemma 26.** *Let C be a sufficiently large constant, and let $t \geq \frac{C \log n}{\log(d/\lambda)}$. Then for every $u \in V \setminus B$,*

$$\|\bar{A}^t \mathbb{1}_u\|_2^2 = \Theta\left(\frac{1}{n}\right) \qquad \textit{and} \qquad \|M^t \mathbb{1}_u\|_2^2 = \Theta\left(\frac{1}{n}\right).$$

Proof. Recall that $1 = \lambda_1 \geq \dots \geq \lambda_n$ denotes the eigenvalues of $\bar{A}$, and $e_1, \dots e_n$ denotes the corresponding eigenvectors. We have

$$\|\bar{A}^t \mathbb{1}_u\|_2^2 = \sum_{i=1}^n \lambda_i^{2t} \langle \mathbb{1}_u, e_i \rangle^2 = \frac{\langle \mathbb{1}_u, D^{1/2}\mathbb{1} \rangle^2}{2m} + \sum_{i=2}^n \lambda_i^{2t} \langle \mathbb{1}_u, e_i \rangle^2 = \frac{\deg(u)}{2m} + \sum_{i=2}^n \lambda_i^{2t} \langle \mathbb{1}_u, e_i \rangle^2 \tag{8}$$

where the second transition uses $\lambda_1 = 1$ and $e_1 = \frac{D^{1/2}\mathbb{1}}{\sqrt{2m}}$, and the last transition uses $\langle \mathbb{1}_u, D^{1/2}\mathbb{1}\rangle^2 = \deg(u)$. By Observation 15, $m = O(nd)$ and by the model assumption (Definition 6), $\deg(u) = \Omega(d)$. Since every term in Equation (8) is non-negative, this gives $\|\bar{A}^t\mathbb{1}_u\|_2^2 \geq \frac{\deg(u)}{2m} = \Omega\left(\frac{1}{n}\right)$.

We now prove that $\|\bar{A}^t\mathbb{1}_u\|_2^2 = O(1/n)$. By Lemma 16, for every $i \in [2, n-k+1]$, $|\lambda_i| \leq C'\sqrt{\lambda/d}$, for some constant C'. By the assumption that C is sufficiently large, we can assume that $t \geq \frac{C\log n}{\log(d/\lambda)} \geq \frac{\log n}{\log(\sqrt{d/\lambda}/C')}$. This gives $\lambda_i^{2t} \leq \left(C'\sqrt{\lambda/d}\right)^{2t} \leq n^{-2}$. Substituting this into Equation (8) gives

$$\begin{aligned}
\|\bar{A}^t\mathbb{1}_u\|_2^2 &= \frac{\deg(u)}{2m} + \sum_{i=2}^{n-k+1}\lambda_i^{2t}\langle \mathbb{1}_u, e_i\rangle^2 + \sum_{i=n-k+2}^{n}\lambda_i^{2t}\langle \mathbb{1}_u, e_i\rangle^2 \\
&\leq \frac{\deg(u)}{2m} + \frac{1}{n^2}\sum_{i=2}^{n-2}\langle \mathbb{1}_u, e_i\rangle^2 + \sum_{i=n-k+2}^{n}\lambda_i^{2t}\langle \mathbb{1}_u, e_i\rangle^2 \\
&\leq \frac{\deg(u)}{2m} + \frac{1}{n^2}\|\mathbb{1}_u\|_2^2 + \|f_u\|_2^2 \\
&= O\left(\frac{1}{n}\right).
\end{aligned}$$

Here, the last transition uses that $\|f_u\|_2^2 = O(1/n)$ by Lemma 21, since $u \in V \setminus B$, and that $\deg(u) \leq d$ and $m = \Omega(nd)$, by Observation 15.

The guarantee on $\|M^t\mathbb{1}_u\|_2^2$ follows from $M^t\mathbb{1}_u = D^{1/2}\bar{A}^tD^{-1/2}\mathbb{1}_u$, together with the assumption that $\deg(v) = \Theta(d)$ for all $v \in V$ (see Observation 15). ◀

We are now ready to prove the correctness of SpectralDotProduct(Lemma 10).

Proof of Lemma 10. Let C be a sufficiently large constant, let $\delta > 0$ be a sufficiently small constant. Let $t_{\min} \coloneqq \frac{10C\log n}{\log(d/\lambda)}$, and let $p(x) = x^{t_{\min}}q(x)$ be the polynomial from Theorem 4 applied with parameters δ, C and $t = t_{\min}$. Let $t_\Delta = \deg q = O\left(\sqrt{\lambda/d}\log n\right)$. For a sufficiently large constant κ, let

$$r = \kappa\sqrt{n}\cdot(\max|\mathrm{coeff}(p)|)^4(t_\Delta+1)^6. \tag{9}$$

By Theorem 4, $\max|\mathrm{coeff}(p)| = n^{O\left(\frac{1}{\log(d/\lambda)}\right)}$, so $r = \widetilde{O}\left(\sqrt{n}\cdot n^{O\left(\frac{1}{\log(d/\lambda)}\right)}\right)$.

We now show that with probability at least 0.9, the quantity computed by the algorithm approximates $\sum_{t,t'} c_t, c_{t'}\langle \bar{A}_\star\mathbb{1}_u, \bar{A}_\star\mathbb{1}_v\rangle = \langle p(\bar{A}_\star)\mathbb{1}_u, p(\bar{A}_\star)\mathbb{1}_v\rangle$ to a high precision.

▷ Claim 27. With probability at least 0.9, it holds that

$$\left|\sqrt{\deg(u)}\sqrt{\deg(v)}\sum_{t,t'}c_tc_{t'}\left(\left\langle D^{-1/2}\hat{p}_u^t, D^{-1/2}\hat{p}_v^{t'}\right\rangle - \frac{1}{2m}\right) - \langle p(\bar{A}_\star)\mathbb{1}_u, p(\bar{A}_\star)\mathbb{1}_v\rangle\right| \leq \frac{1}{160}\frac{k}{n}.$$

Proof. Fix $t, t' \in [t_{\min}, t_{\min}+t_\Delta]$. Set $p = M^t\mathbb{1}_u$, $q = M^{t'}\mathbb{1}_v$ and $W = D^{-1}$ in Lemma 25. Then $p^\top Wq = \langle D^{-1/2}M^t\mathbb{1}_u, D^{-1/2}M^{t'}\mathbb{1}_v\rangle$ and $\|p\|_2^2 + \|q\|_2^2 = \|M^t\mathbb{1}_u\|_2^2 + \|M^{t'}\mathbb{1}_v\|_2^2 = \Theta(1/n)$, where the last equality follows by Lemma 26. Therefore, applying Lemma 25 with $w_{\max} = O(1/d)$, $\zeta = \frac{0.1}{(t_\Delta+1)^2}$, $\xi = \frac{\gamma}{c_tc_{t'}(t_\Delta+1)^2}$ for a sufficiently small constant γ, and r as in Equation (9), we get that with probability at least $1 - \frac{0.1}{(t_\Delta+1)^2}$, it holds that

$$\left|\left\langle D^{-1/2}\hat{p}_u^t, D^{-1/2}\hat{p}_v^{t'}\right\rangle - \langle D^{-1/2}M^t\mathbb{1}_u, D^{-1/2}M^{t'}\mathbb{1}_v\rangle\right| \leq \frac{1}{160|c_t||c_{t'}|(t_\Delta+1)^2}\cdot\frac{k}{dn}$$

By taking a union bound over the $(t_\Delta+1)^2$ possible pairs of $t,t' \in [t_{\min}, t_{\min}+t_\Delta]$, we get

$$\Pr\left[\left|\sum_{t,t'} c_t c_{t'}\left(\left\langle D^{-1/2}\hat{p}_u^t, D^{-1/2}\hat{p}_v^{t'}\right\rangle - \langle D^{-1/2}M^t\mathbb{1}_u, D^{-1/2}M^{t'}\mathbb{1}_v\rangle\right)\right| \le \frac{k}{160dn}\right] \ge 0.9. \quad (10)$$

From Lemma 24, we have

$$\begin{aligned}
&\sum_{t,t'} c_t c_{t'}\left(\left\langle D^{-1/2}\hat{p}_u^t, D^{-1/2}\hat{p}_v^{t'}\right\rangle - \langle D^{-1/2}M^t\mathbb{1}_u, D^{-1/2}M^{t'}\mathbb{1}_v\rangle\right)\\
=&\sum_{t,t'} c_t c_{t'}\left(\left\langle D^{-1/2}\hat{p}_u^t, D^{-1/2}\hat{p}_v^{t'}\right\rangle - \frac{1}{2m}\right) - \sum_{t,t'} c_t c_{t'}\left(\langle D^{-1/2}M^t\mathbb{1}_u, D^{-1/2}M^{t'}\mathbb{1}_v\rangle - \frac{1}{2m}\right)\\
=&\sum_{t,t'} c_t c_{t'}\left(\left\langle D^{-1/2}\hat{p}_u^t, D^{-1/2}\hat{p}_v^{t'}\right\rangle - \frac{1}{2m}\right) - \frac{1}{\sqrt{\deg(u)}\sqrt{\deg(v)}}\sum_{t,t'} c_t c_{t'}\langle \bar{A}_\star^t\mathbb{1}_u, \bar{A}_\star^{t'}\mathbb{1}_v\rangle\\
=&\sum_{t,t'} c_t c_{t'}\left(\left\langle D^{-1/2}\hat{p}_u^t, D^{-1/2}\hat{p}_v^{t'}\right\rangle - \frac{1}{2m}\right) - \frac{1}{\sqrt{\deg(u)}\sqrt{\deg(v)}}\langle p(\bar{A}_\star)\mathbb{1}_u, p(\bar{A}_\star)\mathbb{1}_v\rangle.
\end{aligned}$$

Combining with Equation (10) gives the claim. ◁

Finally, by Lemma 23, since $u,v \in V\setminus B$, we have

$$\left|\langle p(\bar{A}_\star)\mathbb{1}_u, p(\bar{A}_\star)\mathbb{1}_v\rangle - \langle f_u, f_v\rangle\right| \le \frac{1}{160}\frac{k}{n}. \quad (11)$$

Combining Claim 27 and Equation (11) gives

$$\Pr\left[\left|\textsc{SpectralDotProduct}(u,v) - \langle f_u, f_v\rangle\right| \le \frac{1}{80}\frac{k}{n}\right] \ge 0.9.$$

Running time. The running time of SpectralDotProduct is dominated by the $2(t_\Delta+1) = \widetilde{O}(1)$ calls to RandomWalks (Algorithm 1) in lines 5-6, and the $(t_\Delta+1)^2 = \widetilde{O}(1)$ inner product computations in line 8. Each call to RandomWalks$(r,t,\cdot)$ takes time $O(t\cdot r) = \widetilde{O}(r)$ and returns a vector $\hat{p}_u^t$ of support at most r. Then, each of the inner product computations $\langle D^{-1/2}\hat{p}_u^t, D^{-1/2}\hat{p}_v^{t'}\rangle$ in line 8 can be performed in time

$$O\left(\min\left\{|\mathrm{supp}(D^{-1/2}\hat{p}_u^t)|, |\mathrm{supp}(D^{-1/2}\hat{p}_v^{t'})|\right\}\right) = O(r)$$

by computing $\sum_{w\in\mathrm{supp}(\hat{p}_u^t)\cap\mathrm{supp}(\hat{p}_v^{t'})}\hat{p}_u^t(w)\hat{p}_v^{t'}(w)/\deg(w)$. So the overall running time is $\widetilde{O}(r) = \widetilde{O}\left(n^{1/2+O\left(\frac{1}{\log(d/\lambda)}\right)}\right)$. ◀

5.3 Analysis of SpectralDistance (Proof of Lemma 11 and Corollary 12)

In this section, we prove Lemma 11 and Corollary 12. We start by proving Lemma 11.

Proof of Lemma 11. Let $i_{\max} = 5\cdot 10^6 \log n$ denote the number of iterations in SpectralDistance (Algorithm 3). Let $u,v \in V\setminus B$. For every $i \in [1, i_{\max}]$, each of the calls to SpectralDotProduct in lines 2, 3, 4 succeeds with probability at least 0.9. Therefore, by Lemma 10 and the union bound, with probability at least 0.7, it holds that

$$\begin{aligned}
&\left|\|f_u - f_v\|^2_{\mathrm{apx},i} - \|f_u - f_v\|_2^2\right| \\
&\quad = \left|\left(\|f_u\|^2_{\mathrm{apx},i} + \|f_v\|^2_{\mathrm{apx},i} - 2\langle f_u, f_v\rangle_{\mathrm{apx},i}\right) - \left(\|f_u\|_2^2 + \|f_v\|_2^2 - 2\langle f_u, f_v\rangle\right)\right| \\
&\quad \le \left|\|f_u\|^2_{\mathrm{apx},i} - \|f_u\|_2^2\right| + \left|\|f_v\|^2_{\mathrm{apx},i} - \|f_v\|_2^2\right| + 2\left|\langle f_u, f_v\rangle_{\mathrm{apx},i} - \langle f_u, f_v\rangle\right| \\
&\quad \le \frac{k}{80n} + \frac{k}{80n} + \frac{2k}{80n} = \frac{k}{20n}.
\end{aligned}$$

Therefore, by a Chernoff bound, for each pair $u, v \in V \setminus B$,

$$\Pr\left[\left|\underset{i\in[1,i_{\max}]}{\mathrm{median}}\left(\|f_u - f_v\|^2_{\mathrm{apx},i}\right) - \|f_u - f_v\|_2^2\right| \ge \frac{k}{20n}\right] \le \exp(-0.00025 i_{\max}) \le n^{-102},$$

where the last inequality follows from the setting $i_{\max} = 5 \cdot 10^6 \log n$. Taking a union bound over all possible pairs of $u, v \in V \setminus B$ (of which there are at most n^2) gives the required guarantee.

Running time. By Lemma 10, each call to SpectralDotProduct takes time $\widetilde{O}\left(n^{1/2+O\left(\frac{1}{\log(d/\lambda)}\right)}\right)$. Since the total number of calls to SpectralDotProduct is $O(\log n) = \widetilde{O}(1)$, we get the claimed running time. ◀

Next, we prove Corollary 12.

Proof of Corollary 12. Condition on the event that for all $u, v \in V \setminus B$,

$$\left|\textsc{SpectralDistance}(u, v) - \|f_u - f_v\|_2^2\right| \le \frac{k}{20n}. \tag{12}$$

By Lemma 11, this happens with probability at least $1 - n^{-100}$. Suppose that $\chi(u) = \chi(v) = i$. Then by the triangle inequality and Definition 19,

$$\|f_u - f_v\|_2 \le \|f_u - \mu_i\|_2 + \|f_v - \mu_i\|_2 \le 2 \cdot \sqrt{\frac{0.05k}{n}} = \sqrt{\frac{0.2k}{n}}. \tag{13}$$

By Equations (12) and (13), we get

$$\begin{aligned}
&\textsc{SpectralDistance}(u, v) \le \left|\textsc{SpectralDistance}(u, v) - \|f_u - f_v\|_2^2\right| + \|f_u - f_v\|_2^2 \\
&\le \frac{0.05k}{n} + \frac{0.2k}{n} = \frac{k}{4n}.
\end{aligned}$$

Suppose instead that $\chi(u) = i \neq \chi(v)$. Then by the triangle inequality and Definition 19,

$$\|f_u - f_v\|_2 \ge \|f_v - \mu_i\|_2 - \|f_u - \mu_i\|_2 \ge \sqrt{\frac{k}{n}} - \sqrt{\frac{0.05k}{n}} \ge \sqrt{\frac{0.56k}{n}}. \tag{14}$$

By Equation (12) and (14) we get

$$\begin{aligned}
&\textsc{SpectralDistance}(u, v) \ge \|f_u - f_v\|_2^2 - \left|\textsc{SpectralDistance}(u, v) - \|f_u - f_v\|_2^2\right| \\
&\ge \frac{0.56k}{n} - \frac{0.05k}{n} > \frac{k}{2n}.
\end{aligned}$$

◀

5.4 Construction of the polynomial (Proof of Theorem 4)

In this section, we prove Theorem 4. More generally, we prove:

▶ **Theorem 28.** *Let $\alpha \in [-1,1]$ be a non-zero constant (independent of n), and let $\delta > 0$. For every $\epsilon > 0$ smaller than a sufficiently small constant (depending on $|\alpha|$ and δ), and every $\frac{10\log n}{\log(1/\epsilon)} \le t \le O\left(\frac{\log n}{\log(1/\epsilon)}\right)$, there exists a polynomial p of the form*

$$p(x) = x^t q(x)$$

where q is a polynomial of degree $O(\epsilon \log n)$ such that the coefficients of p are bounded in absolute value by $n^{O_{\alpha,\delta}\left(\frac{1}{\log(1/\epsilon)}\right)}$ and

1. *$|p(x) - 1| \le \delta$ for $x \in [\alpha - \epsilon, \alpha + \epsilon]$*
2. *$|p(x)| \le n^{-2}$ for $|x| \le \epsilon$.*

As a corollary of Theorem 28, we obtain Theorem 4.

Proof of Theorem 4. Immediate from Theorem 28 by setting $\alpha = \frac{-1}{(k-1)}$ and $\epsilon = C\sqrt{\lambda/d}$. ◀

For the rest of the section, we prove Theorem 28. Our polynomial builds on the polynomial construction from [22], and we use the following two results.

▶ **Theorem 29** (Theorem A.2 in [22])**.** *Let $t \ge 2$, $\epsilon \in (0, 0.1]$, $d \ge 10\epsilon t$, and let*

$$P(x) = \sum_{v=0}^{d} T_v(x) c_{v,\epsilon,t}$$

be the polynomial obtained by taking the first $d+1$ terms of the Chebyshev expansion of $(1-\epsilon x)^{-t}$ on $[-1,1]$, where T_v denotes the polynomial given by $T_v(\cos\theta) = \cos(v\theta)$. Then

- *$\max_{v \le d} |c_{v,\epsilon,t}| \le 2^{O(\epsilon t)}$, and*
- *$\sup_{x\in[-1,1]} |P(x) - (1-\epsilon x)^{-t}| = O(de^{-d})$.*

▷ Claim 30 (Claim A.3 in [22]). There exists an absolute constant C such that for every $\epsilon, t > 0$ and every $d \ge 10\epsilon t$, the maximum coefficient of the polynomial $P\left(\frac{1-x}{\epsilon}\right)$ is bounded in absolute value by $(1/\epsilon)^{C\cdot d}$, where $P(y)$ denotes the degree d polynomial from Theorem 29.

Proof of Theorem 28. The main idea is to split the proof into the two cases $\epsilon \lesssim |\alpha|/\log n$, in which case the polynomial $p(x) = (x/\alpha)^t$ is already enough, and the case $\epsilon \gtrsim |\alpha|/\log n$, in which case we will use the polynomial from Theorem 29.

To keep the coefficients small enough in the first case, we choose $t \approx \frac{\log n}{\log(1/\epsilon)}$ rather than $t \approx \log n$, as in [22]. We then analyze the second case more carefully for this choice of t, instead of relying directly on the analysis in [22].

First, we need to set up some parameters. Let C_{29} denote the hidden universal constant in Theorem 29 such that $\sup_{x\in[-1,1]} |P(x) - (1-\epsilon x)^{-t}| \le C_{29} de^{-d}$ for all ϵ. Suppose that ϵ satisfies

$$\log(1/\epsilon) \ge \frac{\log(C_{29}/\delta)}{\delta} \tag{15}$$

(this holds by the assumption that ϵ is smaller than a sufficiently small constant). Let $\tau \ge 10$ be the constant such that

$$t = \frac{\tau \log n}{\log(1/\epsilon)}.$$

Additionally, define the constant

$$\kappa := \frac{\log(C_{29}/\delta)}{4\tau} \leq \frac{\log(C_{29}/\delta)}{40}. \tag{16}$$

We now split the proof into the two cases $\epsilon \leq \frac{\kappa|\alpha|}{\log n}$ or $\epsilon > \frac{\kappa|\alpha|}{\log n}$.

Case 1. Suppose $\epsilon \leq \frac{\kappa|\alpha|}{\log n}$. Let $p(x) = \frac{x^t}{\alpha^t}$. Then clearly p is of the right form, and the only non-zero coefficient of p is bounded in absolute value by

$$\left(\frac{1}{|\alpha|}\right)^t = \left(\frac{1}{|\alpha|}\right)^{\frac{\tau \log n}{\log(1/\epsilon)}} = n^{\frac{\tau \log(1/\alpha)}{\log(1/\epsilon)}} = n^{O_\alpha\left(\frac{1}{\log(1/\epsilon)}\right)},$$

by the assumption that $t = \frac{\tau \log n}{\log(1/\epsilon)}$ for some constant τ.

We now verify Condition 1. By the assumption $\epsilon \leq \frac{\kappa|\alpha|}{\log n}$, we have

$$\begin{aligned}
\frac{4\epsilon t}{|\alpha|} &= \frac{4\tau\epsilon \log n}{|\alpha| \log(1/\epsilon)} && \text{by the definition of } \tau \\
&\leq \frac{4\tau\kappa}{\log(1/\epsilon)} && \text{by the assumption } \epsilon \leq \frac{\kappa|\alpha|}{\log n} \\
&\leq \delta && \text{by Equation (16) and Equation (15).}
\end{aligned}$$

Therefore,

$$\max_{x \in [\alpha-\epsilon, \alpha+\epsilon]} |p(x) - 1| = \max_{x \in [\alpha-\epsilon, \alpha+\epsilon]} \left| \left(\frac{x}{\alpha}\right)^t - 1 \right| \leq \exp\left(\frac{2t\epsilon}{|\alpha|}\right) - 1 \leq \frac{4t\epsilon}{|\alpha|} \leq \delta.$$

Next, we verify Condition 2. We have

$$\max_{x \in [-\epsilon, \epsilon]} |p(x)| = \max_{x \in [-\epsilon, \epsilon]} \left| \left(\frac{x}{\alpha}\right)^t \right| = \left(\frac{\epsilon}{|\alpha|}\right)^t \leq \exp\left(-t \log(|\alpha|/\epsilon\right) \leq n^{-2},$$

where the third transition uses the assumption $t \geq \frac{10 \log n}{\log(1/\epsilon)}$, and the last transition holds for n sufficiently large, using the assumption α is a constant independent of n.

Case 2. Suppose $\epsilon \geq \frac{\kappa|\alpha|}{\log n}$. Let $P(x)$ be the polynomial from Theorem 29 applied with error parameter $\epsilon/|\alpha|$ and $d = \frac{10 \log(1/\epsilon)\epsilon t}{|\alpha|} = \frac{10\tau\epsilon \log n}{|\alpha|} = O_\alpha(\epsilon \log n)$, and let $p(x) := \left(\frac{x}{\alpha}\right)^t P\left(\frac{1 - \frac{x}{\alpha}}{\frac{\epsilon}{|\alpha|}}\right)$. We start by bounding the coefficients of $p(x)$. By Claim 30 and the setting of d, the coefficients of the polynomial $q(x) := x^t P\left(\frac{1-x}{\frac{\epsilon}{|\alpha|}}\right)$ are bounded in absolute value by $(|\alpha|/\epsilon)^{C_{30} \cdot d} = n^{O_\alpha(\epsilon \log(1/\epsilon))} = n^{O_\alpha(1/\log(1/\epsilon))}$, where C_{30} denotes the universal constant from Claim 30. Since $p(x) = q(x/\alpha)$, the coefficients of p are bounded in absolute value by

$$\begin{aligned}
\max |\text{coeff}(p)| &\leq \left(\frac{1}{|\alpha|}\right)^{\deg p} \cdot \max |\text{coeff}(q)| = \left(\frac{1}{|\alpha|}\right)^{t+d} n^{O_\alpha\left(\frac{1}{\log(1/\epsilon)}\right)} \\
&\leq \left(\frac{1}{|\alpha|}\right)^{\frac{\tau \log n}{\log(1/\epsilon)} + \frac{10\tau\epsilon \log n}{|\alpha|}} n^{O_\alpha\left(\frac{1}{\log(1/\epsilon)}\right)} = n^{O_\alpha\left(\frac{1}{\log(1/\epsilon)}\right)},
\end{aligned}$$

as required. We now prove that Condition 1 holds. We have

$$\begin{aligned}
&\max_{x\in[\alpha-\epsilon,\alpha+\epsilon]} |p(x)-1| \\
&= \max_{x\in[\alpha-\epsilon,\alpha+\epsilon]} \left|\left(\frac{x}{\alpha}\right)^t P\left(\frac{1-x/\alpha}{\epsilon/|\alpha|}\right) - 1\right| \\
&\le \left(1+\frac{\epsilon}{|\alpha|}\right)^t \max_{x\in[\alpha-\epsilon,\alpha+\epsilon]} \left|P\left(\frac{1-x/\alpha}{\epsilon/|\alpha|}\right) - \left(\frac{x}{\alpha}\right)^{-t}\right| \\
&\le C_{29}\cdot\left(1+\frac{\epsilon}{|\alpha|}\right)^t de^{-d} && \text{by Theorem 29} \\
&\le C_{29}\cdot\exp\left(\frac{\epsilon t}{|\alpha|}\right)\exp\left(-\frac{d}{2}\right) && \text{since } x\le e^{x/2} \text{ and } 1+x\le e^x \\
&= C_{29}\cdot\exp\left(\frac{\epsilon t}{|\alpha|} - 5\log(1/\epsilon)\frac{\epsilon}{|\alpha|}t\right) && \text{by the setting } d = 10\log(1/\epsilon)\epsilon t/|\alpha| \\
&\le C_{29}\cdot\exp\left(-4\log(1/\epsilon)\frac{\epsilon}{|\alpha|}t\right) \\
&\le C_{29}\exp\left(-4\kappa\tau\right) && \text{by } t=\frac{\tau\log n}{\log(1/\epsilon)} \text{ and } \epsilon\ge\frac{\kappa|\alpha|}{\log n} \\
&= \delta && \text{by setting } \kappa \text{ (16).}
\end{aligned}$$

We now prove Condition 2. Since p has at most $(d+1)$ non-zero coefficients, we have

$$\begin{aligned}
&\max_{x\in[-\epsilon,\epsilon]} |p(x)| \le \epsilon^t\cdot(d+1)\cdot\max|\mathrm{coeff}(p)| \\
&\le \epsilon^{10\log n/\log(1/\epsilon)}\cdot O\left(\epsilon\log n/|\alpha|\right)\cdot n^{O_\alpha(1/\log(1/\epsilon))} \le n^{-10}\cdot n\cdot n^{O_\alpha(1/\log(1/\epsilon))} \le n^{-2},
\end{aligned}$$

where the last inequality uses that ϵ is smaller than a sufficiently small constant depending on α, so we can assume that the value of the exponent $O_\alpha(1/\log(1/\epsilon))$ is at most 7. ◀

5.5 Putting everything together (Proof of Theorem 1)

We now combine Theorem 4, Lemma 11, Corollary 12 and Lemma 20 to obtain Theorem 1.

Proof of Theorem 1. The oracle runs the preprocessing algorithm FindRepresentatives (Algorithm 5) to obtain a set of k color representatives $\{c_i\}_{i\in[k]}$. Then, for every query u, the oracle returns $\textsc{Color}(u,\{c_i\}_{i\in[k]})$.

Let B be the set of bad vertices from Definition 19. We now show that with probability at least 0.9, the oracle is correct on all vertices in $V\setminus B$.

By Lemma 14, with probability at least 0.99, the output of FindRepresentatives is a valid set of color representatives (as per Definition 13). Claim 31 below shows that conditioned on the success of FindRepresentatives, the oracle $\textsc{Color}(v,\{c_i\}_{i\in[k]})$ returns the correct color (up to a permutation of colors) for every $v\in V\setminus B$:

▷ Claim 31. Suppose that $\{c_i\}_{i\in[k]}$ is a set of color representatives (as per Definition 13). Let π be the permutation such that $\pi(\chi(c_i))=i$ for every $i\in[k]$. Then with probability at least $1-n^{-100}$, for all $v\in V\setminus B$,

$$\textsc{Color}(v,\{c_i\}_{i\in[k]}) = \pi(\chi(v)).$$

Proof. Let $v \in V \setminus B$, and let c_i be the color representative such that $\chi(c_i) = \chi(v)$. Since $\{c_j\}_{j\in[k]}$ is a set of color representatives, by Definition 13, every c_j belongs to a different color class, and $c_j \in V \setminus B$ for all $j \in [k]$. Therefore, by Corollary 12,

$$\textsc{SpectralDistance}(v, c_i) \leq \frac{k}{4n}$$

and for all $j \neq i$,

$$\textsc{SpectralDistance}(v, c_j) \geq \frac{k}{2n}.$$

In particular, $\arg\min_j \textsc{SpectralDistance}(v, c_j) = i$. By definition of the permutation π, we have $i = \pi(\chi(c_i)) = \pi(\chi(v))$, so Color returns $\pi(\chi(v))$ as claimed. ◁

Taking a union bound over the success of FindRepresentatives and Claim 31, we get that with probability at least $0.99 - n^{-100} \geq 0.9$, every vertex in $V \setminus B$ is colored correctly (up to a permutation of the colors). In that case, by Lemma 20,

$$|\{v : \textsc{Color}(v, \{c_i\}_{i\in[k]}) \neq \pi(\chi(v))\}| \leq |B| \leq O\left(\sqrt{\frac{\lambda}{d}} \cdot n\right).$$

Running time. The preprocessing stage runs FindRepresentatives (Algorithm 5). By Lemma 14, this has running time $\widetilde{O}\big(n^{1/2+O(1/\log(d/\lambda))}\big)$.

Each query is answered by Color (Algorithm 4), which calls SpectralDistance (Algorithm 3) $k = O(1)$ times. By Lemma 11, this takes time $\widetilde{O}\big(n^{1/2+O(1/\log(d/\lambda))}\big)$. ◀

References

1. Noga Alon and Nabil Kahalé. A spectral technique for coloring random 3-colorable graphs. *SIAM J. Comput.*, 26(6):1733–1748, 1997. doi:10.1137/S0097539794270248.
2. Sanjeev Arora and Eden Chlamtac. New approximation guarantee for chromatic number. In Jon M. Kleinberg, editor, *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 215–224. ACM, 2006. doi:10.1145/1132516.1132548.
3. Sanjeev Arora and Rong Ge. New tools for graph coloring. In Leslie Ann Goldberg, Klaus Jansen, R. Ravi, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 14th International Workshop, APPROX 2011, and 15th International Workshop, RANDOM 2011, Princeton, NJ, USA, August 17-19, 2011. Proceedings*, volume 6845 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2011. doi:10.1007/978-3-642-22935-0_1.
4. Mitali Bafna, Jun-Ting Hsieh, and Pravesh K. Kothari. Rounding large independent sets on expanders. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, STOC '25, pages 631–642, New York, NY, USA, 2025. Association for Computing Machinery. doi:10.1145/3717823.3718137.
5. Bonnie Berger and John Rompel. A better performance guarantee for approximate graph coloring. *Algorithmica*, 5(3):459–466, 1990. doi:10.1007/BF01840398.
6. A. Blum and J. Spencer. Coloring random and semi-random k-colorable graphs. *Journal of Algorithms*, 19(2):204–234, 1995. doi:10.1006/jagm.1995.1034.
7. Avrim Blum. New approximation algorithms for graph coloring. *J. ACM*, 41(3):470–516, 1994. doi:10.1145/176584.176586.
8. Avrim Blum and David R. Karger. An õ(n^{3/14})-coloring algorithm for 3-colorable graphs. *Inf. Process. Lett.*, 61(1):49–53, 1997. doi:10.1016/S0020-0190(96)00190-1.

9 Rares-Darius Buhai, Yiding Hua, David Steurer, and Andor Vári-Kakas. Finding colorings in one-sided expanders. *CoRR*, abs/2508.02825, 2025. doi:10.48550/arXiv.2508.02825.

10 Ashish Chiplunkar, Michael Kapralov, Sanjeev Khanna, Aida Mousavifar, and Yuval Peres. Testing graph clusterability: Algorithms and lower bounds. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 497–508. IEEE, 2018. doi:10.1109/FOCS.2018.00054.

11 Artur Czumaj, Pan Peng, and Christian Sohler. Testing cluster structure of graphs. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 723–732. ACM, 2015. doi:10.1145/2746539.2746618.

12 Roee David and Uriel Feige. On the effect of randomness on planted 3-coloring models. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 77–90, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2897518.2897561.

13 Irit Dinur, Elchanan Mossel, and Oded Regev. Conditional hardness for approximate coloring. In *Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '06, pages 344–353, New York, NY, USA, 2006. Association for Computing Machinery. doi:10.1145/1132516.1132567.

14 Uriel Feige and Eran Ofek. Spectral techniques applied to sparse random graphs. *Random Struct. Algorithms*, 27(2):251–275, September 2005. doi:10.1002/RSA.20089.

15 Hendrik Fichtenberger, Michael Kapralov, Ekaterina Kochetkova, Silvio Lattanzi, Davide Mazzali, and Weronika Wrzos-Kaminska. Spectral clustering with side information. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 4327–4341, 2026. doi:10.1137/1.9781611978971.159.

16 J. Friedman, J. Kahn, and E. Szemerédi. On the second eigenvalue of random regular graphs. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, STOC '89, pages 587–598, New York, NY, USA, 1989. Association for Computing Machinery. doi:10.1145/73007.73063.

17 Z. Füredi and J. Komlós. The eigenvalues of random symmetric matrices. *Combinatorica*, 1(3):233–241, 1981. doi:10.1007/BF02579329.

18 M. R. Garey, David S. Johnson, and Larry J. Stockmeyer. Some simplified np-complete graph problems. *Theor. Comput. Sci.*, 1(3):237–267, 1976. doi:10.1016/0304-3975(76)90059-1.

19 Grzegorz Gluch, Michael Kapralov, Silvio Lattanzi, Aida Mousavifar, and Christian Sohler. Spectral clustering oracles in sublinear time. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1598–1617. SIAM, 2021. doi:10.1137/1.9781611976465.97.

20 Jun-Ting Hsieh. Coloring 3-colorable graphs with low threshold rank. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 718–726, 2026. doi:10.1137/1.9781611978971.27.

21 Agastya Vibhuti Jha and Akash Kumar. A sublinear time tester for max-cut on clusterable graphs. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, Tallinn, Estonia, July 8-12, 2024*, volume 297 of *LIPIcs*, pages 91:1–91:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ICALP.2024.91.

22 Michael Kapralov, Ekaterina Kochetkova, and Weronika Wrzos-Kaminska. Spectral clustering in birthday paradox time. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3057–3071, 2026. doi:10.1137/1.9781611978971.113.

23 David R. Karger, Rajeev Motwani, and Madhu Sudan. Approximate graph coloring by semidefinite programming. *J. ACM*, 45(2):246–265, 1998. doi:10.1145/274787.274791.

24 Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972. doi:10.1007/978-1-4684-2001-2_9.

25 Ken-ichi Kawarabayashi and Mikkel Thorup. Coloring 3-colorable graphs with less than $n^{1/5}$ colors. *J. ACM*, 64(1):4:1–4:23, 2017. doi:10.1145/3001582.

26 Ken-ichi Kawarabayashi, Mikkel Thorup, and Hirotaka Yoneda. Better coloring of 3-colorable graphs. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 331–339. ACM, 2024. doi:10.1145/3618260.3649768.

27 Sanjeev Khanna, Nathan Linial, and Shmuel Safra. On the hardness of approximating the chromatic number. *Comb.*, 20(3):393–415, 2000. doi:10.1007/S004930070013.

28 S. Khot. Improved inapproximability results for maxclique, chromatic number and approximate graph coloring. In *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science*, FOCS '01, page 600, USA, 2001. IEEE Computer Society.

29 Akash Kumar, Anand Louis, and Madhur Tulsiani. Finding pseudorandom colorings of pseudorandom graphs. In Satya V. Lokam and R. Ramanujam, editors, *37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2017, Kanpur, India, December 11-15, 2017*, volume 93 of *LIPIcs*, pages 37:1–37:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.FSTTCS.2017.37.

30 A. Nilli. On the second eigenvalue of a graph. *Discrete Math.*, 91(2):207–210, August 1991. doi:10.1016/0012-365X(91)90112-F.

31 Pan Peng and Yuichi Yoshida. Sublinear-time algorithms for max cut, max e2lin (q), and unique label cover on expanders. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 4936–4965. SIAM, 2023. doi:10.1137/1.9781611977554.CH180.

32 Ranran Shen and Pan Peng. A sublinear-time spectral clustering oracle with improved preprocessing time. *Advances in Neural Information Processing Systems*, 36:41809–41828, 2023.

33 Avi Wigderson. Improving the performance guarantee for approximate graph coloring. *J. ACM*, 30(4):729–735, 1983. doi:10.1145/2157.2158.

One-Sided-Error Parameterized Reductions for the Minimum Distance and Shortest Vector Problems*

Shuichi Hirahara
National Institute of Informatics, Tokyo, Japan

Kazuki Ogitsuka
The Graduate University for Advanced Studies (SOKENDAI), Tokyo, Japan

Abstract

It is notoriously difficult to obtain deterministic reductions for the Minimum Distance Problem (MDP) and the Shortest Vector Problem (SVP). Under two-sided-error randomized reductions, Bennett, Cheraghchi, Guruswami, and Ribeiro (STOC 2023) proved parameterized hardness of approximation for these problems. We partially derandomize their reductions and present one-sided-error randomized reductions: MDP is W[1]-hard to approximate within an arbitrary constant factor under FPT many-one one-sided-error randomized reductions; For every fixed $p \geq 1$, SVP in the ℓ_p norm is W[1]-hard to approximate within an arbitrary constant factor below $2^{1/p}$.

We demonstrate the usefulness of one-sided-error randomized reductions by showing that they can be conditionally derandomized when the target problem has an OR function. Under a standard hardness-vs-randomness assumption, namely a plausible lower-bound assumption against nondeterministic circuits, we prove a general theorem formalizing this derandomization. Here, an OR function combines several instances into one instance that preserves their disjunction. We construct such OR functions for the relevant MDP and SVP gap problems, and thereby obtain deterministic W[1]-hardness for approximating MDP over every fixed finite field within every constant factor, and for approximating SVP in ℓ_p norms for every fixed integer p within every factor below $2^{1/p}$. Applying the same framework to Micciancio's one-sided-error randomized reduction (ToC 2012) yields, under the same circuit lower-bound assumption, deterministic polynomial-time NP-hardness of approximating Euclidean SVP within every constant factor.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases Codes, Lattices, Minimum Distance Problem, Shortest Vector Problem, Parameterized complexity, derandomization

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.46

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2608.14305

Funding This work was supported by JSPS KAKENHI Grant Number JP26KJ1224.

1 Introduction

As fundamental computational problems on codes and lattices, the Minimum Distance Problem (MDP) and the Shortest Vector Problem (SVP) have long been studied. In the gap version, γ-GapMDP$_q$ asks whether a linear code over $\mathbb{F}_q$ has a nonzero codeword of Hamming weight at most k, or whether every nonzero codeword has weight larger than γk. Similarly, γ-GapSVP$_p$ asks whether an integer lattice has a nonzero vector of ℓ_p norm at most d, or whether every nonzero vector has norm larger than γd. For further background, see surveys and standard textbooks [3, 31, 35, 41].

* Due to space limitations, some details are omitted from this version.


© Shuichi Hirahara and Kazuki Ogitsuka;
licensed under Creative Commons License CC-BY 4.0



For MDP, Dumer, Micciancio, and Sudan [18] proved randomized NP-hardness for every constant approximation factor. Cheng and Wan [13] derandomized this result, Austrin and Khot [5] and Micciancio [34] gave simpler deterministic reductions. For SVP, van Emde Boas [39] proved NP-hardness of the decision version in the ℓ_∞ norm and conjectured hardness in the Euclidean norm. Ajtai [4] proved randomized NP-hardness of Euclidean SVP. Subsequent work by Khot [27, 28], Haviv and Regev [23], Micciancio [32, 33], and Bennett and Peikert [9] established stronger randomized hardness results for GapSVP$_p$ in several norms and approximation regimes. More recently, Hecht and Safra [24], Hair and Sahai [22, 21], and Wan [40] obtained deterministic hardness results for SVP in finite ℓ_p norms.

Parameterized complexity emerged as a systematic framework in the late 1980s and early 1990s, mainly through the work of Downey and Fellows [15, 16, 17], building on earlier work with Langston; see also [14]. In this framework, the running time of an algorithm is measured not only as a function of the input length n, but also as a function of an additional parameter k. A parameterized problem is *fixed-parameter tractable* if it can be solved in time $f(k) \cdot n^{O(1)}$ for some computable function f. The class of such problems is denoted by FPT. The standard hardness notion used to rule out fixed-parameter tractability is W[1]-hardness, which plays a role analogous to NP-hardness in classical complexity theory. For γ-GapMDP$_q$ and γ-GapSVP$_p$, one of the natural parameters is the distance threshold. Bhattacharyya, Bonnet, Egri, Ghoshal, Karthik C. S., Lin, Manurangsi, and Marx [11] proved that γ-GapMDP$_2$ is W[1]-hard under randomized reductions for every constant $\gamma \geq 1$. They also proved that, for every fixed $p > 1$, γ-GapSVP$_p$ is W[1]-hard for every $\gamma \in [1, (1/2 + 1/2^p)^{-1/p})$, again under randomized reductions. Bennett, Cheraghchi, Guruswami, and Ribeiro [7] extended these parameterized hardness results to GapMDP$_q$ over every fixed finite field and to GapSVP$_p$ in finite ℓ_p norms. More precisely, for every fixed $p > 1$, they proved W[1]-hardness for every constant approximation factor $\gamma \geq 1$. For the remaining case $p = 1$, they proved W[1]-hardness for every $\gamma \in [1, 2)$. Except for binary MDP in [11], these parameterized reductions have two-sided error. In the classical setting, randomized reductions with one-sided error were already used by Micciancio [32, 33] and Bennett and Peikert [9]; Wan [40] subsequently derandomized this line for approximation factors $\gamma \in [1, 2^{1/p})$.

1.1 Our Results

Our first contribution is to strengthen the parameterized hardness reductions of Bennett, Cheraghchi, Guruswami, and Ribeiro [7] from two-sided error to one-sided error.

Our first theorem proves this for γ-GapMDP$_q$ over every fixed finite field, without losing their hardness [7] for every constant approximation factor.

▶ **Theorem 1.** *For every fixed prime power $q \geq 2$ and every constant $\gamma \geq 1$, γ-GapMDP$_q$ is* W[1]*-hard under randomized* FPT *many-one reductions with one-sided error.*

Our second theorem proves the corresponding statement for γ-GapSVP$_p$ in the range $\gamma < 2^{1/p}$.

▶ **Theorem 2.** *For any fixed $p \in [1, \infty)$ and constant $\gamma \in [1, 2^{1/p})$, γ-GapSVP$_p$ is* W[1]*-hard under randomized* FPT *many-one reductions with one-sided error.*

In the classical setting, Micciancio [33] viewed his randomized reduction with one-sided error for the NP-hardness of γ-GapSVP$_2$ as a *partial derandomization*. Following this terminology, Theorems 1 and 2 can be viewed as a partial derandomization of the randomized parameterized hardness reductions of MDP and SVP.

The significance of one-sidedness is that it provides a route to removing the remaining randomness under standard hardness-vs-randomness assumptions. Our second contribution is to show that, for MDP and SVP, the randomized hardness results with one-sided error above can be converted into deterministic hardness results under a standard circuit lower-bound assumption. We assume that there exists an $\varepsilon > 0$ such that

$$\mathsf{E} := \mathsf{DTIME}(2^{O(n)}) \not\subseteq i.o.\text{-}\mathrm{NSIZE}(2^{\varepsilon n}).$$

Here, $\mathrm{NSIZE}(s(n))$ denotes the class of languages decidable by nondeterministic circuits of size $O(s(n))$, and i.o. stands for "infinitely often". Under this assumption, we show in Section 4 that suitable one-sided-error randomized reductions to promise problems with OR functions can be made deterministic. An OR function combines a finite list of instances into a single instance that is YES if at least one input is YES, and is NO if all inputs are NO. In the parameterized setting, we additionally require the output parameter to be bounded by a computable function of the largest input parameter.

To apply this framework to MDP and SVP, we show that the relevant gap problems have the required OR functions.

▶ **Proposition 3** (Informal; see Lemmas 36–38)**.** *Let $\gamma \geq 1$ be a constant. The following hold.*

1. *For every fixed prime power $q \geq 2$, γ-GapMDP_q, parameterized by the distance threshold, has an FPT OR function.*
2. *For every fixed $p \geq 1$, γ-GapSVP_p has a polynomial-time OR function. Moreover, when parameterized by the distance threshold, γ-GapSVP_p has an FPT OR function.*

Combining the general derandomization framework with the OR functions for MDP and SVP yields the following deterministic hardness results. They show that one-sidedness is not merely a stronger error guarantee.

▶ **Theorem 4.** *Assume that there exists an $\varepsilon > 0$ such that*

$$\mathsf{E} \not\subseteq i.o.\text{-}\mathrm{NSIZE}(2^{\varepsilon n}).$$

Then, for every fixed prime power $q \geq 2$ and constant $\gamma \geq 1$, γ-GapMDP_q is $\mathsf{W}[1]$-hard under deterministic FPT many-one reductions.

▶ **Theorem 5.** *Assume that there exists an $\varepsilon > 0$ such that*

$$\mathsf{E} \not\subseteq i.o.\text{-}\mathrm{NSIZE}(2^{\varepsilon n}).$$

For every fixed integer $p \geq 1$ and constant $\gamma \in [1, 2^{1/p})$, γ-GapSVP_p is $\mathsf{W}[1]$-hard under deterministic FPT many-one reductions.[1]

Finally, the conditional derandomization principle also applies to the classical, non-parameterized setting. Starting from Micciancio's randomized hardness result with one-sided error for γ-GapSVP_2 [33], we obtain a deterministic hardness result under the same circuit lower-bound assumption.

▶ **Theorem 6.** *Assume that there exists an $\varepsilon > 0$ such that*

$$\mathsf{E} \not\subseteq i.o.\text{-}\mathrm{NSIZE}(2^{\varepsilon n}).$$

Then, for every fixed constant $\gamma \geq 1$, γ-GapSVP_2 is NP-hard under deterministic polynomial-time many-one reductions.

[1] More generally, the result holds for every fixed $p \geq 1$ such that the language of pairs $(\mathbf{B}, d)$, where $\mathbf{B} \in \mathbb{Z}^{m \times n}$ is a lattice basis, $d \in \mathbb{Z}_{>0}$, and there exists $\boldsymbol{z} \in \mathbb{Z}^n \setminus \{\mathbf{0}\}$ with $\|\mathbf{B}\boldsymbol{z}\|_p \leq d$, belongs to NP.

Compared with Wan's unconditional deterministic polynomial-time many-one hardness result for $1 \leq \gamma < \sqrt{2}$ [40], Theorem 6 extends the approximation range to every constant $\gamma \geq 1$, at the cost of the circuit lower-bound assumption above.

1.2 Technical Overview

The reductions follow Khot's approach [27] used by Bennett, Cheraghchi, Guruswami, and Ribeiro [7]. In that approach, a locally dense object is combined with an inhomogeneous instance, and a random sparsification step is used to make the output homogeneous. The random step has two roles: in YES instances, it should preserve at least one good vector; in NO instances, it should rule out annoying short vectors.

For MDP, we keep the random linear map used for the YES case, but add a deterministic checking block. The block is built from a BCH parity-check matrix $\mathbf{P}$ and a syndrome amplification map $\mathbf{E}$. Every nonzero vector of Hamming weight at most the relevant threshold has a nonzero syndrome under $\mathbf{P}$, and $\mathbf{E}$ expands any nonzero syndrome into more than the allowed number of nonzero coordinates. Thus every potential short vector in a NO instance is ruled out deterministically.

For SVP, the checking block is analogous. We choose a prime q, use a Reed–Solomon parity-check matrix modulo q, and lift it to the integers. A large scaling factor forces any nonzero syndrome in the checking block to have norm above the soundness threshold. The proof then separates possible short lattice vectors according to the coefficient of the target vector and shows that all cases are impossible in NO instances, independently of the remaining randomness.

For the conditional hardness of MDP and SVP, we assume that there exists an $\varepsilon > 0$ such that

$$\mathsf{E} \not\subseteq i.o.\text{-NSIZE}(2^{\varepsilon n}).$$

By the hardness-vs-randomness results of Impagliazzo and Wigderson [26], Klivans and van Melkebeek [29], and Shaltiel and Umans [38], this assumption gives logarithmic-seed pseudorandom generators against polynomial-size nonadaptive SAT-oracle circuits. For a fixed input, let $\phi(r)$ be the predicate that the random tape r makes the randomized reduction output a YES instance. On YES inputs, completeness gives $\Pr[\phi(r) = 1] \geq 2/3$. For the MDP and SVP instances used in our conditional applications, the YES language of the target problem belongs to NP. Thus the success predicate ϕ can be represented by the SAT oracle circuits considered above. The generator obtained from the circuit lower bound has logarithmic seed length and guarantees that, whenever such a predicate ϕ accepts at least a $2/3$ fraction of random tapes, there is a seed σ such that $\phi(G(\sigma)) = 1$. Hence it suffices to enumerate all seeds of G, run the reduction using the corresponding tapes $G(\sigma)$, and combine the outputs by an OR function. On NO inputs, every tape gives a NO instance since the reduction has a one-sided error. For promise problems, an OR function maps a tuple of instances to a single instance that is YES if at least one input is YES, and is NO if all inputs are NO.

1.3 Related Work

Fine-grained hardness of lattice problems and related problems has been studied under assumptions such as ETH, SETH, Gap-ETH, and Gap-SETH. Relevant results include hardness for SVP by Aggarwal and Stephens-Davidowitz [2], hardness for Bounded Distance Decoding (BDD) in ℓ_p norms by Bennett and Peikert [8], improved bounds for BDD and

SVP by Bennett, Peikert, and Tang [10], lower bounds for parameterized Nearest Codeword Problem (NCP) and related problems by Li, Lin, and Liu [30], consequences for parameterized MDP, Closest Vector Problem (CVP), and SVP via earlier gap-preserving reductions [11, 7], and recent ETH-based hardness results for CVP, SVP, and BDD by Aggarwal, Gupta, Morolia, and Zhang [1]. Several of the lattice hardness results cited above use locally dense lattice gadgets, sparsification, or both, but the specific gadgets and reductions in those works differ from ours.

Hittmeir [25] proved deterministic fine-grained hardness results for GapSVP_p in a different parameterized setting, where p is part of the input and also serves as the parameter. In contrast, in this paper p is fixed and the distance threshold is the parameter.

OR functions and related combining reductions have appeared in other contexts. Chang and Kadin [12] studied polynomial-time OR functions for languages, and Ball, Boyle, Degwekar, Deshpande, Rosen, Vaikuntanathan, and Vasudevan [6] studied lossy reductions that combine several input instances according to a Boolean function.

There is also an earlier conditional route to deterministic hardness for SVP: Micciancio [32] proved deterministic many-one NP-hardness of approximating SVP under a number-theoretic conjecture on the distribution of square-free smooth numbers.

1.4 Open Problems

Our results leave several natural open questions. Can the assumption $\mathsf{E} \not\subseteq i.o.\text{-NSIZE}(2^{\varepsilon n})$ for some $\varepsilon > 0$ be removed from Theorems 4 and 5? For SVP, Theorem 2 gives randomized $\mathsf{W}[1]$-hardness with one-sided error for $\gamma < 2^{1/p}$, whereas Bennett, Cheraghchi, Guruswami, and Ribeiro [7] obtained randomized hardness with two-sided error for every constant factor for fixed $p > 1$; can their entire approximation range be made one-sided error? For the Euclidean norm, Wan [40] proved deterministic polynomial-time NP-hardness of γ-GapSVP_2 for every constant $1 \le \gamma < \sqrt{2}$; can deterministic polynomial-time many-one hardness be extended to every constant $\gamma \ge \sqrt{2}$?

2 Preliminaries

2.1 Notations

All vectors are column vectors unless stated otherwise. For a positive integer N, we write $[N] := \{1, \dots, N\}$. We write $\mathbb{Z}_{>0}$ for the set of positive integers. For a finite set S, its cardinality is denoted by $|S|$; for a string x, its bit length is denoted by $|x|$. We use $\langle x_1, \dots, x_t \rangle$ for a standard polynomial-time computable encoding of a tuple. Unless explicitly indicated otherwise, logarithms are base 2; $\log_q$ denotes logarithm with base q. We use $\mathrm{poly}(\cdot)$ for an unspecified polynomial and allow constants hidden in $O(\cdot)$ to depend on fixed constants such as p and q, and on γ whenever γ is fixed throughout the statement. We write U_n for the uniform distribution over $\{0,1\}^n$.

For a map $f\colon X \to Y$ and subset $S \subseteq X$, we write $f(S) := \{f(x) : x \in S\} \subseteq Y$ for the image of S under f. For a set S in an abelian group and an element a, we write $S - a := \{s - a : s \in S\}$.

For $\boldsymbol{x} = (x_1, \dots, x_m)$, its support is $\mathrm{supp}(\boldsymbol{x}) := \{i \in [m] : x_i \neq 0\}$. For vectors over either $\mathbb{F}_q$ or $\mathbb{Z}$, we write $\|\boldsymbol{x}\|_0 := |\mathrm{supp}(\boldsymbol{x})|$. For $\boldsymbol{x} \in \mathbb{R}^m$ and $p \in [1, \infty)$, we write $\|\boldsymbol{x}\|_p := (\sum_{i=1}^m |x_i|^p)^{1/p}$, and $\|\boldsymbol{x}\|_\infty := \max_i |x_i|$. The Hamming distance between two vectors over $\mathbb{F}_q$ is $\|\boldsymbol{x} - \boldsymbol{y}\|_0$. For $r \ge 0$, define the Hamming ball over $\mathbb{F}_q^m$ by $B_{q,m}(r) := \{\boldsymbol{x} \in \mathbb{F}_q^m : \|\boldsymbol{x}\|_0 \le r\}$ and the ℓ_p ball in $\mathbb{R}^m$ by $\mathcal{B}_p^m(r) := \{\boldsymbol{x} \in \mathbb{R}^m : \|\boldsymbol{x}\|_p \le r\}$. When integer vectors are intended, we use $\mathcal{B}_p^m(r) \cap \mathbb{Z}^m$ implicitly.

2.2 Probability Theory

We use a standard consequence of Chebyshev's inequality.

▶ **Lemma 7.** *Let $X_1, \ldots, X_N$ be pairwise independent random variables over $\{0,1\}$ such that $\Pr[X_i = 1] = p > 0$ for $i = 1, \ldots, N$. Then, it holds that*

$$\Pr[\forall i \in [N], X_i = 0] \leq \frac{1}{pN}.$$

2.3 Promise Problems and Reductions

A promise problem is a pair $\Pi = (\Pi_{\text{YES}}, \Pi_{\text{NO}})$ of disjoint languages. A parameterized promise problem is such a pair whose instances are pairs (x, k), where $k \in \mathbb{N}$ is the parameter. For details on promise problems and parameterized complexity, see the survey by Goldreich [19] and the textbooks by Downey and Fellows [17, 16, 15].

Some reductions used are randomized. They are required to preserve NO instances for every choice of randomness, while YES instances need only be mapped correctly with constant probability.

▶ **Definition 8** (Randomized many-one reductions with one-sided error)**.** *Let $\Pi = (\Pi_{\text{YES}}, \Pi_{\text{NO}})$ and $\Pi' = (\Pi'_{\text{YES}}, \Pi'_{\text{NO}})$ be promise problems. A randomized polynomial-time algorithm R is called a randomized many-one reduction with one-sided error from Π to Π' if, on every input x, it outputs a single instance $R(x; r)$ in the input format of Π', and the following hold:*
1. *if $x \in \Pi_{\text{YES}}$, then $\Pr_r[R(x; r) \in \Pi'_{\text{YES}}] \geq 2/3$;*
2. *if $x \in \Pi_{\text{NO}}$, then $\Pr_r[R(x; r) \in \Pi'_{\text{NO}}] = 1$.*

In particular, for YES inputs we allow unsuccessful random strings to produce outputs outside the promise of Π'.

For parameterized problems, we also require that the running time be fixed-parameter tractable and that the output parameter depend only on the input parameter. This gives the parameterized analogue of the preceding randomized reduction.

▶ **Definition 9** (Randomized FPT many-one reductions with one-sided error)**.** *Let $\Pi = (\Pi_{\text{YES}}, \Pi_{\text{NO}})$ and $\Pi' = (\Pi'_{\text{YES}}, \Pi'_{\text{NO}})$ be parameterized promise problems. A randomized algorithm R is called a randomized* FPT *many-one reduction with one-sided error from Π to Π' if there exist a computable function T, a computable function g, and a constant $c > 0$ such that, on input (x, k), the algorithm runs in time at most $T(k) \cdot |x|^c$, outputs a pair (x', k'), and satisfies*
1. *$k' \leq g(k)$;*
2. *if $(x, k) \in \Pi_{\text{YES}}$, then $\Pr[(x', k') \in \Pi'_{\text{YES}}] \geq 2/3$;*
3. *if $(x, k) \in \Pi_{\text{NO}}$, then $\Pr[(x', k') \in \Pi'_{\text{NO}}] = 1$.*

▶ Remark 10. With respect to the fixed encoding of tuples, the running-time bounds

$$T(k) \cdot |x|^c \qquad \text{and} \qquad T'(k) \cdot |\langle x, k \rangle|^c$$

are equivalent up to replacing T by another computable function of k. Therefore, in Definition 9 (and deterministic FPT many-one reduction), one may measure the polynomial factor using either $|x|$ or $|\langle x, k \rangle|$.

We assume that all random choices are implemented using a fixed-length random tape whose length is computable within the stated running-time bound. Standard bounded rejection sampling is used when necessary, and its arbitrarily small loss is absorbed into the completeness probability.

2.4 Coding Problems

For a matrix $\mathbf{G} \in \mathbb{F}_q^{m \times n}$, the linear code generated by the columns of $\mathbf{G}$ is $\mathcal{C} = \mathcal{C}(\mathbf{G}) := \{\mathbf{G}\boldsymbol{x} : \boldsymbol{x} \in \mathbb{F}_q^n\} \subseteq \mathbb{F}_q^m$. The minimum distance of a nonzero linear code $\mathcal{C} \subseteq \mathbb{F}_q^m$ is $\lambda(\mathcal{C}) := \min_{\boldsymbol{c} \in \mathcal{C} \setminus \{\mathbf{0}\}} \|\boldsymbol{c}\|_0$. We use the convention $\lambda(\{\mathbf{0}\}) := +\infty$. For a target vector $\boldsymbol{t} \in \mathbb{F}_q^m$, we write $\mathrm{dist}(\mathcal{C}, \boldsymbol{t}) := \min_{\boldsymbol{c} \in \mathcal{C}} \|\boldsymbol{c} - \boldsymbol{t}\|_0$.

The Minimum Distance Problem (MDP) asks for the Hamming weight of the shortest nonzero codeword in a given linear code. In this paper we use its gap version, where one distinguishes codes of minimum distance at most k from those of minimum distance greater than γk.

▶ **Definition 11** (γ-GapMDP$_q$, [18]). *For $\gamma \geq 1$ and a prime power $q \geq 2$, the (parameterized) promise problem γ-approximate Minimum Distance Problem over $\mathbb{F}_q$ (γ-GapMDP$_q$) is defined as follows. Instances are pairs $(\mathbf{G}, k)$, where $\mathbf{G} \in \mathbb{F}_q^{m \times n}$ is a generator matrix of $\mathcal{C}(\mathbf{G})$, and $k \in \mathbb{Z}_{>0}$ is a distance parameter such that*

1. *$(\mathbf{G}, k)$ is a YES instance if $\lambda(\mathcal{C}(\mathbf{G})) \leq k$;*
2. *$(\mathbf{G}, k)$ is a NO instance if $\lambda(\mathcal{C}(\mathbf{G})) > \gamma k$.*

The parameter of interest is k.

The Nearest Codeword Problem (NCP) is the inhomogeneous analogue of MDP: given a target vector, it asks whether some codeword is close to the target. We will use NCP as the starting point for reductions to MDP.

▶ **Definition 12** (γ-GapNCP$_q$, [18]). *For $\gamma \geq 1$ and a prime power $q \geq 2$, the (parameterized) promise problem γ-approximate Nearest Codeword Problem over $\mathbb{F}_q$ (γ-GapNCP$_q$) is defined as follows. Instances are tuples $(\mathbf{G}, \boldsymbol{t}, k)$, where $\mathbf{G} \in \mathbb{F}_q^{m \times n}$ is a generator matrix of $\mathcal{C}(\mathbf{G})$, $\boldsymbol{t} \in \mathbb{F}_q^m$ is a target vector, and $k \in \mathbb{Z}_{>0}$ is a distance parameter such that*

1. *$(\mathbf{G}, \boldsymbol{t}, k)$ is a YES instance if $\mathrm{dist}(\mathcal{C}(\mathbf{G}), \boldsymbol{t}) \leq k$;*
2. *$(\mathbf{G}, \boldsymbol{t}, k)$ is a NO instance if $\mathrm{dist}(\mathcal{C}(\mathbf{G}), \boldsymbol{t}) > \gamma k$.*

The parameter of interest is k.

The next theorem states the W[1]-hardness of γ-GapNCP$_q$. We will use this result later in the reduction to γ-GapMDP$_q$.

▶ **Theorem 13** ([11, Theorem 5.1, adapted] and [7, Theorem 2.7]). *For any real number $\gamma \geq 1$ and any prime power $q \geq 2$, γ-GapNCP$_q$ is W[1]-hard.*

2.5 Lattice Problems

We now turn to the lattice problems considered in the paper. The lattice analogues of the above coding problems are measured with respect to ℓ_p norms.

For a matrix $\mathbf{B} \in \mathbb{Z}^{m \times n}$, the lattice generated by the columns of $\mathbf{B}$ is $\mathcal{L} = \mathcal{L}(\mathbf{B}) := \{\mathbf{B}\boldsymbol{z} : \boldsymbol{z} \in \mathbb{Z}^n\} \subseteq \mathbb{Z}^m$; that is, $\mathcal{L}(\mathbf{B})$ consists of all integer linear combinations of the columns of $\mathbf{B}$. If the columns of $\mathbf{B}$ are linearly independent, then $\mathbf{B}$ is called a lattice basis of $\mathcal{L}(\mathbf{B})$. For $p \in [1, \infty]$, the ℓ_p length of a shortest nonzero lattice vector is $\lambda_1^{(p)}(\mathcal{L}) := \min_{\boldsymbol{v} \in \mathcal{L} \setminus \{\mathbf{0}\}} \|\boldsymbol{v}\|_p$. For a target vector $\boldsymbol{t} \in \mathbb{R}^m$, we write $\mathrm{dist}_p(\mathcal{L}, \boldsymbol{t}) := \min_{\boldsymbol{v} \in \mathcal{L}} \|\boldsymbol{v} - \boldsymbol{t}\|_p$.

One of the central problems on lattices is the Shortest Vector Problem (SVP), which is to find a shortest non-zero vector in a given lattice. In this paper, we study the promise problem version of the Shortest Vector Problem, which is defined as follows:

▶ **Definition 14** (γ-GapSVP$_p$, [20]). *For $p \geq 1$ and $\gamma \geq 1$, the (parameterized) promise problem γ-approximate Shortest Vector Problem (γ-GapSVP$_p$) is defined as follows. Instances are pairs $(\mathbf{B}, d)$, where $\mathbf{B} \in \mathbb{Z}^{m \times n}$ is a lattice basis of $\mathcal{L}(\mathbf{B})$, and $d \in \mathbb{Z}_{>0}$ is a distance parameter such that*

1. $(\mathbf{B}, d)$ *is a YES instance if* $\lambda_1^{(p)}(\mathcal{L}(\mathbf{B})) \le d$;
2. $(\mathbf{B}, d)$ *is a NO instance if* $\lambda_1^{(p)}(\mathcal{L}(\mathbf{B})) > \gamma d$.

The parameter of interest is d.

We will also use the following classical NP-hardness result for approximate SVP for ℓ_2 norm. This theorem is the non-parameterized hardness input for the deterministic hardness result later in the paper.

▶ **Theorem 15** ([33, Corollary 6.3]). *For any constant* $\gamma \ge 1$, γ-GapSVP$_2$ *is* NP*-hard under randomized many-one reductions with one-sided error.*

The Closest Vector Problem (CVP) is the inhomogeneous analogue of SVP: given a lattice and a target vector, it asks whether the target is close to some lattice vector. For the reduction to GapSVP, we use the following gap variant, whose NO instances require all lattice vectors to be far from every nonzero integer multiple of the target vector.

▶ **Definition 16** (γ-GapCVP$_p$, [7]). *For* $p \ge 1$ *and* $\gamma \ge 1$, *the (parameterized) promise problem* γ*-approximate Closest Vector Problem (*γ-GapCVP$_p$*) is defined as follows. Instances are tuples* $(\mathbf{B}, \boldsymbol{t}, d)$, *where* $\mathbf{B} \in \mathbb{Z}^{m \times n}$ *is a lattice basis of* $\mathcal{L}(\mathbf{B})$, $\boldsymbol{t} \in \mathbb{Z}^m$ *is a target vector, and* $d \in \mathbb{Z}_{>0}$ *is a distance parameter such that*

1. $(\mathbf{B}, \boldsymbol{t}, d)$ *is a YES instance if* $\mathrm{dist}_p(\mathcal{L}(\mathbf{B}), \boldsymbol{t}) \le d$;
2. $(\mathbf{B}, \boldsymbol{t}, d)$ *is a NO instance if* $\mathrm{dist}_p(\mathcal{L}(\mathbf{B}), w\boldsymbol{t}) > \gamma d$ *for all* $w \in \mathbb{Z} \setminus \{0\}$.

The parameter of interest is d.

The following theorem provides the parameterized hardness of this CVP variant. It will be combined with our randomized reduction from CVP to SVP.

▶ **Theorem 17** ([11, Theorem 7.2] and [7, Theorem 2.11]). *For any real numbers* $\gamma \ge 1$ *and* $p \ge 1$, γ-GapCVP$_p$ *is* W[1]*-hard under deterministic* FPT *many-one reductions.*

▶ Remark 18. In [11], the NO instance condition for γ-GapCVP$_p$ is stated as $\mathrm{dist}_p(\mathcal{L}(\mathbf{B}), \boldsymbol{t}) > \gamma d$, but their proof in fact shows that $\mathrm{dist}_p(\mathcal{L}(\mathbf{B}), w\boldsymbol{t}) > \gamma d$ for every nonzero integer $w \in \mathbb{Z} \setminus \{0\}$.

2.6 OR functions

An OR function is a deterministic map that, given a finite tuple of strings, outputs one instance of the same promise problem, with the guarantee that it outputs YES if at least one input is a YES instance and outputs NO if all inputs are NO instances.

▶ **Definition 19** (OR functions of a promise problem). *A promise problem* Π *has a polynomial-time OR function if there is a deterministic polynomial-time algorithm* C *such that, for every tuple* $(x_1, \ldots, x_t)$, *if some* $x_i \in \Pi_{\mathrm{YES}}$, *then* $C(x_1, \ldots, x_t) \in \Pi_{\mathrm{YES}}$, *and if all* $x_i \in \Pi_{\mathrm{NO}}$, *then* $C(x_1, \ldots, x_t) \in \Pi_{\mathrm{NO}}$.

▶ **Definition 20** (FPT OR functions of a parameterized promise problem). *A parameterized promise problem* Π *has an* FPT *OR function if there is a deterministic algorithm* C *and computable functions* T, g *such that, on input* $((x_1, k_1), \ldots, (x_t, k_t))$, *the algorithm runs in time* $T(\kappa)N^{O(1)}$, *where* $\kappa = \max_i k_i$ *and* $N = \sum_i |\langle x_i, k_i \rangle|$, *outputs* (y, ℓ) *with* $\ell \le g(\kappa)$, *and satisfies the same YES/NO OR guarantee as above.*

2.7 Circuits and Pseudorandom Generators

Finally, we recall the circuit classes and pseudorandom generators used in the derandomization argument. A nondeterministic circuit is a Boolean circuit with additional nondeterministic input wires; it accepts an input if some assignment to these wires makes the circuit output 1. For a function $s\colon \mathbb{N} \to \mathbb{N}$, $\mathrm{NSIZE}(s(n))$ denotes the class of languages decided by families of nondeterministic circuits of size $O(s(n))$. We write $L \in i.o.\text{-}\mathrm{NSIZE}(s(n))$ if, for infinitely many input lengths n, there is a nondeterministic circuit C_n of size $O(s(n))$ such that for every $x \in \{0,1\}^n$, C_n accepts x if and only if $x \in L$.

We also use nonadaptive SAT-oracle circuits. We write $\mathrm{SIZE}_{\|}^{\mathsf{SAT}}(s(n))$ for languages decided by families of nonadaptive SAT-oracle circuits of size $O(s(n))$. Nonadaptive means that on every path from an input gate to the output gate there is at most one oracle gate.

▶ **Definition 21** (SAT-circuitizable success predicates). *Let R be a randomized FPT many-one reduction from a parameterized promise problem Π to Π'. We say that R has SAT-circuitizable success predicates if there is a polynomial P such that, for every promised input (x,k), if μ is the number of random bits used by R, $m = |\langle x,k\rangle| + \mu$, and*

$$\phi_{x,k}(r) = 1 \iff R((x,k);r) \in \Pi'_{\mathrm{YES}},$$

then $\phi_{x,k}$ is computable by a circuit in $\mathrm{SIZE}_{\|}^{\mathsf{SAT}}(P(m))$. Extra bits of $r \in \{0,1\}^m$ are ignored.

A pseudorandom generator stretches a short seed into a longer string that is indistinguishable from a uniform random string for a specified class of Boolean circuits.

▶ **Definition 22** (Pseudorandom generators against circuit classes). *Let $\mathcal{C} = \{\mathcal{C}_n\}_{n\geq 1}$ be a family of Boolean circuit classes. A family of functions $G_n : \{0,1\}^{m(n)} \to \{0,1\}^n$ is an $\varepsilon(n)$-PRG for $\mathcal{C}$ if, for every sufficiently large n and every Boolean function $D \in \mathcal{C}_n$,*

$$\left| \Pr_{y \leftarrow U_n}[D(y) = 1] - \Pr_{\sigma \leftarrow U_{m(n)}}[D(G_n(\sigma)) = 1] \right| \leq \varepsilon(n).$$

The next theorem is the derandomization tool used in the paper. Under an exponential circuit lower bound for $\mathsf{E} := \mathsf{DTIME}(2^{O(n)})$ against nondeterministic circuits, it gives logarithmic-seed PRGs that fool polynomial-size nonadaptive SAT-oracle circuits.[2]

▶ **Theorem 23** ([26, 29, 38], see also Theorem 2.8 in [37]). *Assume that there exists an $\varepsilon > 0$ such that $\mathsf{E} \not\subseteq i.o.\text{-}\mathrm{NSIZE}(2^{\varepsilon n})$. Then, for every constant $c > 1$, there exists a constant $a > 1$ such that, for all sufficiently large n, there is a $1/n^c$-PRG*

$$G_n : \{0,1\}^{a \log n} \to \{0,1\}^n$$

for $\mathrm{SIZE}_{\|}^{\mathsf{SAT}}(n^c)$, and G_n is computable in time $\mathrm{poly}(n^c)$.

3 W[1]-hardness with One-Sided Error

In this section, we will show that γ-GapMDP_q and γ-GapSVP_p are $\mathsf{W}[1]$-hard under randomized FPT reductions with one-sided error. As in the prior work [7], this reduction is based on locally dense codes/lattices constructed from BCH and Reed–Solomon codes, together with sparsification; however, by additionally incorporating a gadget based on the minimum distance of the locally dense codes/lattices, it achieves one-sided error.

Locally dense codes and locally dense lattices are defined as follows.

[2] In the proof of Theorem 35, the PRG is used only through the following consequence: if $\phi \in \mathrm{SIZE}_{\|}^{\mathsf{SAT}}(m^c)$ satisfies $\Pr_{r \leftarrow U_m}[\phi(r) = 1] \geq 2/3$, then some pseudorandom output $G_m(\sigma)$ satisfies $\phi(G_m(\sigma)) = 1$. Thus an efficiently enumerable polynomial-size hitting set for the same predicates would suffice.

▶ **Definition 24** (Locally dense codes, [7]). *Fix a real number $\alpha \in (0,1)$, positive integers d, N, m, n, and a prime power q. A (q, α, d, N, m, n)-*locally dense code *is specified by a generator matrix $\mathbf{A} \in \mathbb{F}_q^{m \times n}$ and a target vector $\boldsymbol{s} \in \mathbb{F}_q^m$ with the following properties:*

- $\lambda(\mathcal{C}(\mathbf{A})) > d$.
- $|(\mathcal{C}(\mathbf{A}) - \boldsymbol{s}) \cap B_{q,m}(\alpha d)| \geq N$.

▶ **Definition 25** (Locally dense lattices, [7]). *Fix real numbers $\alpha \in (0,1)$ and $p \geq 1$ and positive integers d, N, m, n. A (p, α, d, N, m, n)-*locally dense lattice *is specified by a basis $\mathbf{A} \in \mathbb{Z}^{m \times n}$ and a target vector $\boldsymbol{s} \in \mathbb{Z}^m$ with the following properties:*

- $\lambda_1^{(p)}(\mathcal{L}(\mathbf{A})) > d$.
- $\left|(\mathcal{L}(\mathbf{A}) - \boldsymbol{s}) \cap \mathcal{B}_p^m(\alpha d)\right| \geq N$.

▶ **Lemma 26.** *Let $T \subseteq \mathbb{F}_q^M \setminus \{0\}$ be a set of pairwise linearly independent vectors, and let $\mathbf{R} \in \mathbb{F}_q^{h \times M}$ be uniformly random. For every $\boldsymbol{a} \in \mathbb{F}_q^h$,*

$$\Pr_{\mathbf{R}}[\exists \boldsymbol{w} \in T : \mathbf{R}\boldsymbol{w} = \boldsymbol{a}] \geq 1 - \frac{q^h}{|T|}.$$

Proof. For each $\boldsymbol{w} \in T$, the indicator of the event $\mathbf{R}\boldsymbol{w} = \boldsymbol{a}$ has expectation q^{-h}. Pairwise linear independence of the vectors in T implies pairwise independence of these indicators. The claim follows from Lemma 7. ◀

3.1 W[1]-hardness of γ-GapMDP$_q$ with One-Sided Error

We will show the W[1]-hardness of GapMDP by giving a randomized FPT reduction with one-sided error from GapNCP to GapMDP.

For this, we use properties of BCH codes.

▶ **Theorem 27** (q-ary BCH codes, [7]). *Fix a prime power q. Then, given integers $m' = q^r - 1$ and $1 \leq d \leq m'$ for some integer r, it is possible to construct in time* poly(m′) *a generator matrix $\mathbf{G}_{\mathrm{BCH}} \in \mathbb{F}_q^{m' \times n'}$ such that $\mathcal{C}_{\mathrm{BCH}} = \mathcal{C}(\mathbf{G}_{\mathrm{BCH}}) \subseteq \mathbb{F}_q^{m'}$ has minimum distance at least d and co-dimension*

$$m' - n' \leq \lceil (d-1)(1 - 1/q) \rceil \log_q(m' + 1).$$

Bennett, Cheraghchi, Guruswami, and Ribeiro [7] constructed the following locally dense code using the BCH code from Theorem 27.

▶ **Theorem 28** ([7, Theorem 3.3]). *Fix a prime power $q \geq 2$ and set $\gamma = 4q$. There exists a randomized algorithm which, given positive integers m and $k \leq m$, runs in time* poly(m) *and outputs with probability at least 0.99 a $(q, \alpha, d, N, m', n')$-locally dense code $(\mathbf{A}, \boldsymbol{s})$, where*

$$m', n' \leq \mathrm{poly}(m),$$
$$d = \gamma k = 4qk,$$
$$\alpha = 1 - \frac{1}{2q},$$
$$N = \frac{(qm)^{2d}}{100},$$

provided that m is sufficiently large compared to q. Moreover, $\lambda(\mathcal{C}(\mathbf{A})) > d$ with probability 1.

In addition to these locally dense codes, by using a random matrix and the parity-check matrix of a BCH code, we obtain a randomized FPT reduction with one-sided error.

▶ **Theorem 29.** *For a fixed prime power $q \geq 2$ and real numbers $\gamma := 4q$ and $\gamma' := \gamma/(\gamma-1) = 4q/(4q-1)$, there exists a randomized* FPT *many-one reduction with one-sided error from γ-GapNCP$_q$ to γ'-GapMDP$_q$.*

Proof. We use the following equivalent formulation of $4q$-GapNCP$_q$. An instance $(\mathbf{G}, \boldsymbol{t}, k)$ consists of a matrix $\mathbf{G} \in \mathbb{F}_q^{m \times n}$, a target $\boldsymbol{t} \in \mathbb{F}_q^m$, and an integer $k \geq 1$. In the NO case we are promised that

$$\operatorname{dist}(\mathcal{C}(\mathbf{G}), \beta \boldsymbol{t}) > 4qk \qquad \text{for every } \beta \in \mathbb{F}_q \setminus \{0\}.$$

This is equivalent to the usual definition because, for every $\beta \in \mathbb{F}_q \setminus \{0\}$,

$$\operatorname{dist}(\mathcal{C}(\mathbf{G}), \beta \boldsymbol{t}) = \min_{\boldsymbol{c} \in \mathcal{C}(\mathbf{G})} \|\boldsymbol{c} - \beta \boldsymbol{t}\|_0 = \min_{\boldsymbol{c}' \in \mathcal{C}(\mathbf{G})} \|\beta \boldsymbol{c}' - \beta \boldsymbol{t}\|_0 = \min_{\boldsymbol{c}' \in \mathcal{C}(\mathbf{G})} \|\boldsymbol{c}' - \boldsymbol{t}\|_0 = \operatorname{dist}(\mathcal{C}(\mathbf{G}), \boldsymbol{t}),$$

since $\mathcal{C}(\mathbf{G})$ is closed under multiplication by nonzero field elements and $\|\beta \boldsymbol{v}\|_0 = \|\boldsymbol{v}\|_0$ for all $\boldsymbol{v} \in \mathbb{F}_q^m$.

By a standard preprocessing, we may assume $1 \leq k \leq m$ and that m is sufficiently large compared with the fixed field size q. Indeed, if $k > m$, the input is automatically a YES instance, and if m is bounded by a constant depending only on q, the instance can be solved exactly by brute-force search. In these exceptional cases we output a fixed YES or NO instance of γ'-GapMDP$_q$ accordingly.

We first describe the reduction. Set $d := \gamma k$, $\alpha := 1 - 1/(2q)$, and $k' := k + \alpha d = (4q-1)k$. By Theorem 28, in randomized polynomial time on input (m, k) we obtain, with probability at least 0.99, a $(q, \alpha, d, N, m', n')$-locally dense code $(\mathbf{A}, \boldsymbol{s})$, where $m', n' \leq \operatorname{poly}(m)$ and $N = (qm)^{2d}/100$. Moreover, Theorem 28 guarantees that $\lambda(\mathcal{C}(\mathbf{A})) > d$ with probability 1. Define

$$\mathcal{S} := (\mathcal{C}(\mathbf{A}) - \boldsymbol{s}) \cap B_{q,m'}(\alpha d).$$

Then, with probability at least 0.99, we have $|\mathcal{S}| \geq (qm)^{2d}/100$. Next define

$$\widehat{m} := \min \left\{ q^r - 1 : q^r - 1 \geq \max\{m, d+1\} \right\}.$$

By Theorem 27 applied with block length $\widehat{m}$ and distance parameter $d+1$, there is a q-ary BCH code of length $\widehat{m}$ and minimum distance at least $d+1$ whose co-dimension h satisfies

$$h \leq \lceil d(1-1/q) \rceil \log_q(\widehat{m}+1) = d(1-1/q) \log_q(\widehat{m}+1).$$

The equality holds because $d(1-1/q) = 4(q-1)k$ is an integer. Let $\mathbf{H}_P \in \mathbb{F}_q^{h \times \widehat{m}}$ be a parity-check matrix of this BCH code. Let $\mathbf{P} \in \mathbb{F}_q^{h \times m}$ be the matrix consisting of the first m columns of $\mathbf{H}_P$. Then every nonzero vector $\boldsymbol{u} \in \mathbb{F}_q^m$ with $\|\boldsymbol{u}\|_0 \leq d$ satisfies $\mathbf{P}\boldsymbol{u} \neq 0$. Indeed, if $\mathbf{P}\boldsymbol{u} = 0$, then the vector obtained from $\boldsymbol{u}$ by padding $\widehat{m} - m$ zeros would be a nonzero codeword of the BCH code checked by $\mathbf{H}_P$ of ℓ_0-norm at most d, contradicting its minimum distance at least $d+1$. Define the linear map

$$\mathbf{E} : \mathbb{F}_q^h \to \mathbb{F}_q^{(d+1)h}$$

by

$$\mathbf{E}(a_1, \ldots, a_h) := (a_1 \mathbf{1}_{d+1}, \ldots, a_h \mathbf{1}_{d+1}),$$

where $\mathbf{1}_{d+1}$ is the all-ones vector of length $d+1$. Then every nonzero $\boldsymbol{a} \in \mathbb{F}_q^h$ satisfies $\|\mathbf{E}\boldsymbol{a}\|_0 \geq d+1$. Finally, sample $\mathbf{R} \in \mathbb{F}_q^{h \times m'}$ uniformly at random, and output the code generated by

$$\mathbf{G}' := \begin{pmatrix} \mathbf{G} & \mathbf{0} & -\boldsymbol{t} \\ \mathbf{0} & \mathbf{A} & -\boldsymbol{s} \\ \mathbf{EPG} & \mathbf{ERA} & -\mathbf{E}(\mathbf{P}\boldsymbol{t} + \mathbf{R}\boldsymbol{s}) \end{pmatrix}$$

together with the parameter k'. Equivalently, for every $(\boldsymbol{x}, \boldsymbol{y}, \beta) \in \mathbb{F}_q^{n+n'+1}$, the corresponding codeword is $\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y}, \beta) = \big(\boldsymbol{u}, \boldsymbol{w}, \mathbf{E}(\mathbf{P}\boldsymbol{u} + \mathbf{R}\boldsymbol{w})\big)$, where $\boldsymbol{u} := \mathbf{G}\boldsymbol{x} - \beta\boldsymbol{t}$ and $\boldsymbol{w} := \mathbf{A}\boldsymbol{y} - \beta\boldsymbol{s}$.

We now verify that this is an FPT reduction. By Theorem 28, we have $m', n' \leq \mathrm{poly}(m)$. Also $\widehat{m} \leq q(\max\{m, d+1\} + 1) = O_q(m)$ because $k \leq m$ and hence $d = 4qk \leq 4qm$. Therefore $h = O_q(k \log m)$, the output length is polynomial in the input size, and the output parameter is $k' = (4q-1)k$. Hence the reduction runs in time $f(k) \cdot \mathrm{poly}(|\mathbf{G}| + |\boldsymbol{t}|)$ and maps k to $O(k)$.

It remains to prove completeness and soundness.

Completeness. Assume that $(\mathbf{G}, \boldsymbol{t}, k)$ is a YES instance of $4q$-GapNCP$_q$. Then there exists $\boldsymbol{x}' \in \mathbb{F}_q^n$ such that $\|\mathbf{G}\boldsymbol{x}' - \boldsymbol{t}\|_0 \leq k$. Let $\boldsymbol{u}' := \mathbf{G}\boldsymbol{x}' - \boldsymbol{t}$ and $\boldsymbol{a} := -\mathbf{P}\boldsymbol{u}' \in \mathbb{F}_q^h$. Condition on the event that $(\mathbf{A}, \boldsymbol{s})$ is locally dense, equivalently that $|\mathcal{S}| \geq (qm)^{2d}/100$. This event has probability at least 0.99.

Choose a subset $\mathcal{T} \subseteq \mathcal{S} \setminus \{\mathbf{0}\}$ containing exactly one representative from each one-dimensional subspace that intersects $\mathcal{S} \setminus \{\mathbf{0}\}$ nontrivially. Then distinct elements of $\mathcal{T}$ are pairwise linearly independent. Moreover, on the event under consideration, $|\mathcal{S}| \geq (qm)^{2d}/100 \geq 2$, where the last inequality follows from the preprocessing. Hence

$$|\mathcal{T}| \geq \frac{|\mathcal{S}| - 1}{q - 1} \geq \frac{|\mathcal{S}|}{2q} \geq \frac{(qm)^{2d}}{200q}.$$

By Lemma 26, the probability that no $\boldsymbol{w} \in \mathcal{T}$ satisfies $\mathbf{R}\boldsymbol{w} = \boldsymbol{a}$ is at most $q^h/|\mathcal{T}|$. The BCH codimension bound gives

$$q^h \leq (\widehat{m} + 1)^{d(1-1/q)} \leq \big(q(4q+3)m\big)^d,$$

and our preprocessing ensures

$$\frac{q^h}{|\mathcal{T}|} \leq 200q \left(\frac{4q+3}{qm}\right)^d \leq 0.01.$$

Hence, conditioned on the event that $(\mathbf{A}, \boldsymbol{s})$ is locally dense, with probability at least 0.99 over $\mathbf{R}$ there exists $\boldsymbol{w} \in \mathcal{T} \subseteq \mathcal{S} \setminus \{\mathbf{0}\}$ such that $\mathbf{R}\boldsymbol{w} = -\mathbf{P}\boldsymbol{u}'$.

Fix such a $\boldsymbol{w}$. Since $\boldsymbol{w} \in \mathcal{S} \subseteq \mathcal{C}(\mathbf{A}) - \boldsymbol{s}$, there exists $\boldsymbol{y} \in \mathbb{F}_q^{n'}$ such that $\boldsymbol{w} = \mathbf{A}\boldsymbol{y} - \boldsymbol{s}$ and $\|\boldsymbol{w}\|_0 \leq \alpha d$. Then $\mathbf{P}\boldsymbol{u}' + \mathbf{R}\boldsymbol{w} = 0$, and therefore $\boldsymbol{c}(\boldsymbol{x}', \boldsymbol{y}, 1) = (\boldsymbol{u}', \boldsymbol{w}, 0)$. It follows that

$$\|\boldsymbol{c}(\boldsymbol{x}', \boldsymbol{y}, 1)\|_0 \leq \|\boldsymbol{u}'\|_0 + \|\boldsymbol{w}\|_0 \leq k + \alpha d = k'.$$

Thus the output instance is a YES instance of γ'-GapMDP$_q$.

Since the event that $(\mathbf{A}, \boldsymbol{s})$ is locally dense has probability at least 0.99 and, conditioned on that event, the probability over $\mathbf{R}$ that some $\boldsymbol{w} \in \mathcal{T}$ satisfies $\mathbf{R}\boldsymbol{w} = -\mathbf{P}\boldsymbol{u}'$ is at least 0.99, we conclude that

$$\Pr[\text{output is a YES instance of } \gamma'\text{-GapMDP}_q] \geq 0.99 \cdot 0.99 > 2/3.$$

Soundness. Assume that $(\mathbf{G}, \boldsymbol{t}, k)$ is a NO instance of $4q$-GapNCP$_q$. We show that for every realization of $(\mathbf{A}, \boldsymbol{s})$ produced above and every choice of $\mathbf{R}$, the output is a NO instance of γ'-GapMDP$_q$.

Let $\boldsymbol{c} = \boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y}, \beta) \in \mathcal{C}(\mathbf{G}') \setminus \{0\}$ be an arbitrary nonzero codeword. Suppose toward a contradiction that $\|\boldsymbol{c}\|_0 \le d$. Writing $\boldsymbol{c} = \big(\boldsymbol{u}, \boldsymbol{w}, \mathbf{E}(\mathbf{P}\boldsymbol{u} + \mathbf{R}\boldsymbol{w})\big)$, where $\boldsymbol{u} = \mathbf{G}\boldsymbol{x} - \beta\boldsymbol{t}$ and $\boldsymbol{w} = \mathbf{A}\boldsymbol{y} - \beta\boldsymbol{s}$, we get $\|\boldsymbol{u}\|_0 + \|\boldsymbol{w}\|_0 \le d$.

We first show that $\beta = 0$. If $\beta \neq 0$, then by the NO promise for $4q$-GapNCP$_q$, $\|\boldsymbol{u}\|_0 = \|\mathbf{G}\boldsymbol{x} - \beta\boldsymbol{t}\|_0 > 4qk = d$, contradicting $\|\boldsymbol{u}\|_0 + \|\boldsymbol{w}\|_0 \le d$.

Thus $\beta = 0$. We next show that $\boldsymbol{w} = 0$. If $\boldsymbol{w} \neq 0$, then $\boldsymbol{w} = \mathbf{A}\boldsymbol{y}$ is a nonzero codeword of $\mathcal{C}(\mathbf{A})$, and hence $\|\boldsymbol{w}\|_0 \ge \lambda(\mathcal{C}(\mathbf{A})) > d$, again contradicting $\|\boldsymbol{u}\|_0 + \|\boldsymbol{w}\|_0 \le d$.

Therefore $\beta = 0$ and $\boldsymbol{w} = 0$, so $\boldsymbol{c} = \big(\boldsymbol{u}, 0, \mathbf{E}(\mathbf{P}\boldsymbol{u})\big)$ with $\boldsymbol{u} = \mathbf{G}\boldsymbol{x}$. Since $\boldsymbol{c} \neq 0$, we must have $\boldsymbol{u} \neq 0$. Also $\|\boldsymbol{u}\|_0 \le d$, so by the defining property of $\mathbf{P}$, $\mathbf{P}\boldsymbol{u} \neq 0$. Hence $\|\mathbf{E}(\mathbf{P}\boldsymbol{u})\|_0 \ge d + 1$, which implies $\|\boldsymbol{c}\|_0 \ge d + 1$, contradicting $\|\boldsymbol{c}\|_0 \le d$. This contradiction proves that every nonzero codeword of $\mathcal{C}(\mathbf{G}')$ has ℓ_0-norm strictly larger than d. Therefore $\lambda(\mathcal{C}(\mathbf{G}')) > d$.

Finally, since $d = \gamma' k'$, we obtain $\lambda(\mathcal{C}(\mathbf{G}')) > \gamma' k'$, so the output is a NO instance of γ'-GapMDP$_q$. This holds for every realization of $(\mathbf{A}, \boldsymbol{s})$ and every choice of $\mathbf{R}$, and thus the reduction has one-sided error. ◀

To amplify to an arbitrary approximation constant factor, we use the following well-known fact.

▶ **Fact 30.** *For any two linear codes $\mathcal{C}(\mathbf{G}_1)$ and $\mathcal{C}(\mathbf{G}_2)$ with $\mathbf{G}_1 \in \mathbb{F}_q^{m_1 \times n_1}$, $\mathbf{G}_2 \in \mathbb{F}_q^{m_2 \times n_2}$, and minimum distances $\lambda(\mathcal{C}(\mathbf{G}_1)) = d_1$, $\lambda(\mathcal{C}(\mathbf{G}_2)) = d_2$, we have*

$$\lambda(\mathcal{C}(\mathbf{G}_1 \otimes \mathbf{G}_2)) = d_1 \cdot d_2,$$

where $\mathbf{G}_1 \otimes \mathbf{G}_2 \in \mathbb{F}_q^{m_1 m_2 \times n_1 n_2}$ is the Kronecker product of $\mathbf{G}_1$ and $\mathbf{G}_2$.

By combining the above claims, we obtain the desired proof.

Proof of Theorem 1. By Theorems 13 and 29, the problem

$$\gamma_0\text{-GapMDP}_q \qquad \text{with} \qquad \gamma_0 := \frac{4q}{4q-1} > 1$$

is W[1]-hard under randomized FPT many-one reductions with one-sided error.

Fix an integer $c \ge 1$. Given an instance $(\mathbf{G}, k)$ of γ_0-GapMDP$_q$, map it to $(\mathbf{G}^{\otimes c}, k^c)$, where $\mathbf{G}^{\otimes c}$ denotes the c-fold Kronecker product of $\mathbf{G}$ with itself. By Fact 30, $\lambda(\mathcal{C}(\mathbf{G}^{\otimes c})) = \lambda(\mathcal{C}(\mathbf{G}))^c$. Hence, if $(\mathbf{G}, k)$ is a YES instance, then

$$\lambda(\mathcal{C}(\mathbf{G}^{\otimes c})) \le k^c,$$

so $(\mathbf{G}^{\otimes c}, k^c)$ is a YES instance of γ_0^c-GapMDP$_q$. If $(\mathbf{G}, k)$ is a NO instance, then

$$\lambda(\mathcal{C}(\mathbf{G}^{\otimes c})) > (\gamma_0 k)^c = \gamma_0^c k^c,$$

so $(\mathbf{G}^{\otimes c}, k^c)$ is a NO instance of γ_0^c-GapMDP$_q$. Because c is a constant, this is a deterministic FPT many-one reduction. Therefore, for every integer $c \ge 1$, the problem γ_0^c-GapMDP$_q$ is W[1]-hard under randomized FPT many-one reductions with one-sided error.

Now let $\gamma \ge 1$ be arbitrary. Choose c so that $\gamma_0^c \ge \gamma$. Every YES instance of γ_0^c-GapMDP$_q$ is also a YES instance of γ-GapMDP$_q$, and every NO instance of γ_0^c-GapMDP$_q$ is also a NO instance of γ-GapMDP$_q$. Hence the identity map is a deterministic FPT many-one reduction from γ_0^c-GapMDP$_q$ to γ-GapMDP$_q$. Therefore γ-GapMDP$_q$ is W[1]-hard under randomized FPT many-one reductions with one-sided error. ◀

3.2 W[1]-hardness of γ-GapSVP$_p$ with One-Sided Error

In this section, we prove a randomized FPT reduction with one-sided error from CVP to SVP using ideas similar to those for the hardness of MDP.

▶ **Lemma 31** (Reed–Solomon codes, [36]). *Let q be a prime and let m, r be integers with $1 \le r \le m < q$. Then one can construct in time* $\mathrm{poly(m, \log q)}$ *a matrix $\mathbf{P} \in \mathbb{F}_q^{r\times m}$ such that every nonzero vector $\boldsymbol{u} \in \mathbb{F}_q^m$ with $\|\boldsymbol{u}\|_0 \le r$ satisfies $\mathbf{P}\boldsymbol{u} \neq 0$.*

Proof. Take the standard $r \times m$ Vandermonde parity-check matrix of a Reed–Solomon code over $\mathbb{F}_q$. Any set of at most r columns is linearly independent, so no nonzero vector of Hamming weight at most r lies in its kernel. This construction is deterministic and polynomial time. ◀

The following theorem gives a randomized algorithm for constructing locally dense lattices based on Reed–Solomon codes.

▶ **Theorem 32** ([7, Theorem 4.3]). *Fix real numbers $p \ge 1$ and $\gamma' \in [1, 2^{1/p})$. Let*

$$\varepsilon = (\gamma')^{-p} - \frac{1}{2} > 0,\ \gamma = \left\lceil \max\left(\frac{12}{\varepsilon}, \frac{1}{(1+\varepsilon/2)^{1/p}-1}\right)\right\rceil,\ \alpha = \left(\frac{1}{(\gamma')^p} - \frac{2}{\gamma^p}\right)^{1/p}.$$

Then there exists a randomized algorithm which, on input positive integers m and d, runs in time poly(m, d) and outputs with probability at least 0.99 a $(p, \alpha, \gamma d, N, m', n')$-locally dense lattice $(\mathbf{A}, \boldsymbol{s})$, where

$$m', n' \le \mathit{poly}(m, d),\ N = (2m(1+\gamma d))^{3(\gamma d)^p},$$

provided that m is sufficiently large compared to p. Moreover, $\lambda_1^{(p)}(\mathcal{L}(\mathbf{A})) > \gamma d$ with probability 1.

As in the case of MDP, by using a random matrix and the parity-check matrix of a Reed–Solomon code, we obtain a randomized FPT reduction with one-sided error.

▶ **Theorem 33.** *Fix a real number $p \ge 1$ and a rational constant $\gamma' \in [1, 2^{1/p})$. Define*

$$\varepsilon = (\gamma')^{-p} - \frac{1}{2} > 0 \ \text{ and } \ \gamma = \left\lceil \max\left(\frac{12}{\varepsilon}, \frac{1}{(1+\varepsilon/2)^{1/p}-1}\right)\right\rceil.$$

Then there exists a randomized FPT *many-one reduction with one-sided error from γ-*GapCVP$_p$ *to γ'-*GapSVP$_p$.

Proof. Let $(\mathbf{B}, \boldsymbol{t}, d)$ be an instance of γ-GapCVP$_p$ with $\mathbf{B} \in \mathbb{Z}^{m\times n}$, $\boldsymbol{t} \in \mathbb{Z}^m$, and $d \in \mathbb{Z}_{>0}$.

By padding $\mathbf{B}$ and $\boldsymbol{t}$ with zero rows, if necessary, we may assume that m is sufficiently large compared with p, so that Theorem 32 applies. This padding preserves $\mathrm{dist}_p(\mathcal{L}(\mathbf{B}), a\boldsymbol{t})$ for every $a \in \mathbb{Z}$ and leaves the parameter d unchanged.

Write $\gamma' = s_0/t_0$ in lowest terms with $s_0, t_0 \in \mathbb{Z}_{>0}$. Set

$$\alpha := \left(\frac{1}{(\gamma')^p} - \frac{2}{\gamma^p}\right)^{1/p},\ r := \min\{m, \lceil(\gamma d)^p\rceil\},\ L := s_0,\ d' := \gamma t_0 d = L\frac{\gamma}{\gamma'}d \in \mathbb{Z}_{>0}.$$

By Theorem 32, in randomized time $\mathrm{poly}(m, d)$ on input (m, d) we obtain, with probability at least 0.99, a $(p, \alpha, \gamma d, N, m', n')$-locally dense lattice $(\mathbf{A}, \boldsymbol{s})$, where

$$N = (2m(1+\gamma d))^{3(\gamma d)^p},\ m', n' \le \mathrm{poly}(m, d), \ \text{ and } \ \lambda_1^{(p)}(\mathcal{L}(\mathbf{A})) > \gamma d.$$

Next, choose a prime q satisfying

$$2\max\{m, \gamma d\} < q \leq 4\max\{m, \gamma d\}.$$

By Bertrand's postulate and primality testing, such a prime can be found deterministically in time $\mathrm{poly}(m, d)$. If $\lceil (\gamma d)^p \rceil \geq m$, let $\mathbf{P} := I_m \in \mathbb{F}_q^{m \times m}$. Otherwise, apply Lemma 31 with parameters (q, m, r) to obtain $\mathbf{P} \in \mathbb{F}_q^{r \times m}$ such that every nonzero vector $\boldsymbol{u} \in \mathbb{F}_q^m$ with $\|\boldsymbol{u}\|_0 \leq r$ satisfies $\mathbf{P}\boldsymbol{u} \neq 0$. Let $\widehat{\mathbf{P}} \in \mathbb{Z}^{r \times m}$ be the canonical integer lift of $\mathbf{P}$ whose entries lie in $\{0, 1, \ldots, q-1\}$. Define the reduction modulo q maps

$$\pi_m : \mathbb{Z}^m \to \mathbb{F}_q^m, \qquad \pi_m(\boldsymbol{u}) = \boldsymbol{u} \bmod q,$$
$$\pi_{m'} : \mathbb{Z}^{m'} \to \mathbb{F}_q^{m'}, \qquad \pi_{m'}(\boldsymbol{w}) = \boldsymbol{w} \bmod q.$$

Observe that every nonzero $\boldsymbol{u} \in \mathbb{Z}^m$ with $\|\boldsymbol{u}\|_p \leq \gamma d$ satisfies $\mathbf{P}\pi_m(\boldsymbol{u}) \neq 0$. Indeed, $|u_i| \leq \|\boldsymbol{u}\|_p \leq \gamma d < q/2$ for every coordinate, so $\pi_m(\boldsymbol{u}) \neq 0$. Moreover,

$$\|\pi_m(\boldsymbol{u})\|_0 \leq \|\boldsymbol{u}\|_0 \leq \|\boldsymbol{u}\|_p^p \leq (\gamma d)^p \leq r$$

if $r < m$, while if $r = m$ then $\mathbf{P} = I_m$ and the claim is immediate. Now sample $\mathbf{R} \in \mathbb{F}_q^{r \times m'}$ uniformly at random, let $\widehat{\mathbf{R}} \in \mathbb{Z}^{r \times m'}$ be its canonical integer lift. More precisely, $\mathbf{R}$ is obtained by sampling the entries independently and uniformly at random from $\mathbb{F}_q$. Define the lattice basis

$$\mathbf{B}' := \begin{pmatrix} \mathbf{B} & \mathbf{0}_{m \times n'} & -\boldsymbol{t} & \mathbf{0}_{m \times r} \\ \mathbf{0}_{m' \times n} & \mathbf{A} & -\boldsymbol{s} & \mathbf{0}_{m' \times r} \\ \mathbf{0}_{1 \times n} & \mathbf{0}_{1 \times n'} & 1 & \mathbf{0}_{1 \times r} \\ \beta\widehat{\mathbf{P}}\mathbf{B} & \beta\widehat{\mathbf{R}}\mathbf{A} & -\beta(\widehat{\mathbf{P}}\boldsymbol{t} + \widehat{\mathbf{R}}\boldsymbol{s}) & \beta q \mathbf{I}_r \end{pmatrix}, \qquad \beta := \gamma d + 1.$$

And set $\widehat{\mathbf{B}} := L\mathbf{B}'$. Output the lattice basis $\widehat{\mathbf{B}}$ together with the threshold d'. Equivalently, for every $(\boldsymbol{x}, \boldsymbol{y}, a, \boldsymbol{z}) \in \mathbb{Z}^{n+n'+1+r}$, the corresponding lattice vector is

$$\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y}, a, \boldsymbol{z}) = \big(\boldsymbol{u}, \boldsymbol{w}, a, \beta(\widehat{\mathbf{P}}\boldsymbol{u} + \widehat{\mathbf{R}}\boldsymbol{w} + q\boldsymbol{z})\big),$$

where $\boldsymbol{u} := \mathbf{B}\boldsymbol{x} - a\boldsymbol{t}$ and $\boldsymbol{w} := \mathbf{A}\boldsymbol{y} - a\boldsymbol{s}$. This is the output instance of γ'-GapSVP$_p$.

We now verify that this is an FPT reduction. Since $m', n' \leq \mathrm{poly}(m, d)$ and $r \leq \lceil (\gamma d)^p \rceil$, the output basis has size polynomial in the input size, and the output parameter is $d' = \gamma t_0 d = O(d)$, because γ and t_0 are constants depending only on the fixed target factor γ'. Hence the reduction runs in time $f(d) \cdot \mathrm{poly}(|\mathbf{B}| + |\boldsymbol{t}|)$ and maps d to $O(d)$.

It remains to prove completeness and soundness.

Completeness. Assume that $(\mathbf{B}, \boldsymbol{t}, d)$ is a YES instance of γ-GapCVP$_p$. Then there exists $\boldsymbol{x}' \in \mathbb{Z}^n$ such that $\|\mathbf{B}\boldsymbol{x}' - \boldsymbol{t}\|_p \leq d$. Let

$$\boldsymbol{u}' := \mathbf{B}\boldsymbol{x}' - \boldsymbol{t}, \qquad \boldsymbol{a} := -\mathbf{P}\pi_m(\boldsymbol{u}') \in \mathbb{F}_q^r.$$

Define

$$\mathcal{W}_s := (\mathcal{L}(\mathbf{A}) - \boldsymbol{s}) \cap \mathcal{B}_p^{m'}(\alpha\gamma d).$$

Then, with probability at least 0.99, we have $|\mathcal{W}_s| \geq N$. Because every $\boldsymbol{w} \in \mathcal{W}_s$ has $\|\boldsymbol{w}\|_p \leq \alpha\gamma d < q/2$, the map $\pi_{m'}$ is injective on $\mathcal{W}_s$, and $\pi_{m'}(\boldsymbol{w}) \neq 0$ whenever $\boldsymbol{w} \neq 0$.

Choose $\mathcal{T} \subseteq \mathcal{W}_s \setminus \{\mathbf{0}\}$ so that $\pi_{m'}(\mathcal{T})$ contains exactly one representative from each one-dimensional subspace of $\mathbb{F}_q^{m'}$ that intersects $\pi_{m'}(\mathcal{W}_s \setminus \{\mathbf{0}\})$. Then $\pi_{m'}(\mathcal{T})$ is pairwise linearly independent and, since $|\mathcal{W}_s| \geq N$ and $N \geq q$,

$$|\mathcal{T}| \geq \frac{|\mathcal{W}_s| - 1}{q - 1} \geq \frac{N}{q}.$$

By Lemma 26, the probability that no $\boldsymbol{w} \in \mathcal{T}$ satisfies

$$\mathbf{R}\pi_{m'}(\boldsymbol{w}) = -\mathbf{P}\pi_m(\boldsymbol{u}') \tag{1}$$

is at most $q^r/|\mathcal{T}| \leq q^{r+1}/N$. With $M := 2m(1 + \gamma d)$, the choices of q, r, N give

$$\frac{q^{r+1}}{N} \leq \frac{(2M)^{(\gamma d)^p + 2}}{M^{3(\gamma d)^p}} < 0.01.$$

Thus, conditioned on the locally dense lattice event, with probability at least 0.99 over $\mathbf{R}$ there exists $\boldsymbol{w} \in \mathcal{T} \subseteq \mathcal{W}_s$ satisfying Equation (1).

Fix such a vector $\boldsymbol{w}$. Since $\boldsymbol{w} \in \mathcal{W}_s \subseteq \mathcal{L}(\mathbf{A}) - \boldsymbol{s}$, there exists $\boldsymbol{y} \in \mathbb{Z}^{n'}$ such that $\boldsymbol{w} = \mathbf{A}\boldsymbol{y} - \boldsymbol{s}$ and $\|\boldsymbol{w}\|_p \leq \alpha\gamma d$. Since $\widehat{\mathbf{P}}\boldsymbol{u}' + \widehat{\mathbf{R}}\boldsymbol{w} = 0 \pmod q$, this implies that there exists $\boldsymbol{z} \in \mathbb{Z}^r$ such that

$$\widehat{\mathbf{P}}\boldsymbol{u}' + \widehat{\mathbf{R}}\boldsymbol{w} + q\boldsymbol{z} = 0.$$

Hence $\boldsymbol{c}(\boldsymbol{x}', \boldsymbol{y}, 1, \boldsymbol{z}) = (\boldsymbol{u}', \boldsymbol{w}, 1, \mathbf{0}_r)$. Its ℓ_p norm satisfies

$$\begin{aligned}
\|\boldsymbol{c}(\boldsymbol{x}', \boldsymbol{y}, 1, \boldsymbol{z})\|_p^p &= \|\boldsymbol{u}'\|_p^p + \|\boldsymbol{w}\|_p^p + 1 \\
&\leq d^p + (\alpha\gamma d)^p + 1 \\
&\leq 2d^p + \alpha^p\gamma^p d^p \\
&= (2 + \alpha^p\gamma^p)\, d^p \\
&= \left(\frac{\gamma}{\gamma'}\right)^p d^p,
\end{aligned}$$

where the last equality uses the definition of α. Hence

$$\|\boldsymbol{c}(\boldsymbol{x}', \boldsymbol{y}, 1, \boldsymbol{z})\|_p \leq \frac{\gamma}{\gamma'} d.$$

Therefore the nonzero vector $L\boldsymbol{c}(\boldsymbol{x}', \boldsymbol{y}, 1, \boldsymbol{z}) \in \mathcal{L}(\widehat{\mathbf{B}})$ satisfies

$$\|L\boldsymbol{c}(\boldsymbol{x}', \boldsymbol{y}, 1, \boldsymbol{z})\|_p \leq L\frac{\gamma}{\gamma'} d = d'.$$

Thus the output instance is a YES instance of γ'-GapSVP$_p$. Since the event that $(\mathbf{A}, \boldsymbol{s})$ is $(p, \alpha, \gamma d, N, m', n')$-locally dense lattice has probability at least 0.99 and, conditioned on that event, the probability over $\mathbf{R}$ that some $\boldsymbol{w} \in \mathcal{W}_s$ satisfies the congruence in Equation (1) is at least 0.99, we conclude that

$$\Pr[\text{output is a YES instance of } \gamma'\text{-GapSVP}_p] \geq 0.99 \cdot 0.99 > \frac{2}{3}.$$

Soundness. Assume that $(\mathbf{B}, \boldsymbol{t}, d)$ is a NO instance of γ-GapCVP$_p$. We first show that, regardless of the choice of $\mathbf{R}$, every nonzero vector of $\mathcal{L}(\mathbf{B}')$ has ℓ_p norm strictly larger than γd. Let $\boldsymbol{c} = \boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y}, a, \boldsymbol{z}) \in \mathcal{L}(\mathbf{B}') \setminus \{0\}$ be an arbitrary nonzero lattice vector. Suppose toward a contradiction that $\|\boldsymbol{c}\|_p \le \gamma d$. Writing

$$\boldsymbol{c} = \big(\boldsymbol{u}, \boldsymbol{w}, a, \beta(\widehat{\mathbf{P}}\boldsymbol{u} + \widehat{\mathbf{R}}\boldsymbol{w} + q\boldsymbol{z})\big),$$

where $\boldsymbol{u} = \mathbf{B}\boldsymbol{x} - a\boldsymbol{t}$ and $\boldsymbol{w} = \mathbf{A}\boldsymbol{y} - a\boldsymbol{s}$, we get

$$\|\boldsymbol{u}\|_p^p + \|\boldsymbol{w}\|_p^p + |a|^p \le (\gamma d)^p.$$

In particular, if $a \neq 0$, then by the NO promise for γ-GapCVP$_p$,

$$\|\boldsymbol{u}\|_p = \|\mathbf{B}\boldsymbol{x} - a\boldsymbol{t}\|_p > \gamma d,$$

a contradiction. Hence $a = 0$. If now $\boldsymbol{w} \neq 0$, then because $\lambda_1^{(p)}(\mathcal{L}(\mathbf{A})) > \gamma d$, we get $\|\boldsymbol{w}\|_p > \gamma d$, again a contradiction. Therefore $a = 0$ and $\boldsymbol{w} = 0$. So every short vector must be of the form

$$\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y}, a, \boldsymbol{z}) = (\boldsymbol{u}, 0, 0, \beta(\widehat{\mathbf{P}}\boldsymbol{u} + q\boldsymbol{z})).$$

If $\boldsymbol{u} = 0$, then since $\boldsymbol{c} \neq 0$, $\boldsymbol{z} \neq 0$. Hence

$$\|\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y}, a, \boldsymbol{z})\|_p \ge \beta q > \gamma d,$$

a contradiction. So we may assume $\boldsymbol{u} \neq 0$. Since $\|\boldsymbol{u}\|_p \le \gamma d$, property of $\mathbf{P}$ implies $\mathbf{P}\pi_m(\boldsymbol{u}) \neq 0$. Equivalently,

$$\widehat{\mathbf{P}}\boldsymbol{u} + q\boldsymbol{z} \neq 0 \qquad \text{for every } \boldsymbol{z} \in \mathbb{Z}^r.$$

Therefore the last block of $\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y}, a, \boldsymbol{z})$ is nonzero, so

$$\|\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y}, a, \boldsymbol{z})\|_p \ge \beta > \gamma d,$$

a contradiction. We conclude that every nonzero vector of $\mathcal{L}(\mathbf{B}')$ has ℓ_p norm strictly larger than γd. Since $\mathcal{L}(\widehat{\mathbf{B}}) = L\mathcal{L}(\mathbf{B}')$, every nonzero vector of $\mathcal{L}(\widehat{\mathbf{B}})$ has ℓ_p norm strictly larger than $L\gamma d = \gamma' d'$. Therefore the output is a NO instance of γ'-GapSVP$_p$. This holds for every pair $(\mathbf{A}, \boldsymbol{s})$ from Theorem 32 and every choice of $\mathbf{R}$, and thus the reduction has one-sided error. ◀

From the above, we immediately obtain Theorem 2.

Proof of Theorem 2. Let $\gamma \in [1, 2^{1/p})$ be the target approximation factor. Choose any rational number η such that $\gamma < \eta < 2^{1/p}$. By Theorems 17 and 33, the problem η-GapSVP$_p$ is W[1]-hard under randomized FPT many-one reductions with one-sided error.

Now observe that every YES instance of η-GapSVP$_p$ is also a YES instance of γ-GapSVP$_p$, and every NO instance of η-GapSVP$_p$ is also a NO instance of γ-GapSVP$_p$, because $\eta > \gamma$. Hence any algorithm for γ-GapSVP$_p$ would also solve η-GapSVP$_p$ on the same input.

Therefore γ-GapSVP$_p$ is W[1]-hard under randomized FPT many-one reductions with one-sided error. ◀

4 Deterministic Hardness under the Circuit Lower Bound

In this section, we describe the derandomization principle used later. The proof is the same in the ordinary and parameterized settings: enumerate a small hitting set of random tapes, run the randomized reduction with one-sided error on all of them, and combine the outputs by an OR function.

▶ **Theorem 34.** *Assume that there exists an $\varepsilon > 0$ such that*

$$\mathsf{E} \not\subseteq i.o.\text{-NSIZE}(2^{\varepsilon n}).$$

Let Π and Π' be promise problems. Suppose that
1. *Π is NP-hard under deterministic polynomial-time many-one reductions;*
2. *$\Pi'_{\mathrm{YES}} \in \mathsf{NP}$;*
3. *Π' has a polynomial-time OR function;*
4. *there exists a randomized many-one reduction with one-sided error from Π to Π'.*

Then Π' is NP-hard under deterministic polynomial-time many-one reductions.

Proof sketch. Let R be the randomized reduction. For a fixed input x, let $\phi_x(r) = 1$ iff $R(x; r) \in \Pi'_{\mathrm{YES}}$. Since $\Pi'_{\mathrm{YES}} \in \mathsf{NP}$, after hardwiring x, the predicate ϕ_x is computable by a polynomial-size nonadaptive SAT-oracle circuit: compute $R(x; r)$, reduce membership in Π'_{YES} to SAT, and ask one oracle query.

On YES inputs, completeness gives $\Pr_r[\phi_x(r) = 1] \geq 2/3$. By Theorem 23, the corresponding hitting set contains a tape r with $\phi_x(r) = 1$, once the combined input-and-randomness length is large enough. For the finitely many smaller lengths, enumerate all random tapes. The deterministic reduction runs R on all tapes in the chosen list and combines the resulting instances by the polynomial-time OR function. Completeness follows because one output is YES. Soundness follows because, on NO inputs, one-sidedness ensures that every tape produces a NO instance. Composing with the deterministic hardness reduction to Π proves the theorem. ◀

▶ **Theorem 35.** *Assume that there exists an $\varepsilon > 0$ such that*

$$\mathsf{E} \not\subseteq i.o.\text{-NSIZE}(2^{\varepsilon n}).$$

Let Π and Π' be parameterized promise problems. Suppose that
1. *Π is $\mathsf{W}[1]$-hard under deterministic FPT many-one reductions;*
2. *Π' has an FPT OR function;*
3. *there exists a randomized FPT many-one reduction with one-sided error $R : \Pi \to \Pi'$;*
4. *R has SAT-circuitizable success predicates.*

Then Π' is $\mathsf{W}[1]$-hard under deterministic FPT many-one reductions.

Proof. Let R be the randomized reduction with one-sided error and let C be an FPT OR function for Π'. By the SAT-circuitizability assumption, there is a polynomial P such that every success predicate

$$\phi_{x,k}(r) = 1 \iff R((x, k); r) \in \Pi'_{\mathrm{YES}}$$

is in $\mathrm{SIZE}^{\mathsf{SAT}}_{\|}(P(m))$, where $m = |\langle x, k\rangle| + \mu$ and μ is the number of random bits used by R. Choose a constant c with $P(m) \leq m^c$ for all sufficiently large m, and take the hitting sets $H_m := \{G_m(\sigma) : \sigma \in \{0,1\}^{a \log m}\}$ from Theorem 23. Increase the threshold m_0 if necessary so that the theorem applies for all $m \geq m_0$.

The deterministic reduction is as follows. On input (x, k), compute $m = |\langle x, k\rangle| + \mu$. If $m < m_0$, list all 2^μ random tapes; this is only a constant-size list because m_0 is fixed. If $m \geq m_0$, enumerate H_m. Run R on (x, k) using each listed tape, and apply the FPT OR function C to all resulting outputs.

The running time is FPT: the hitting set has size $m^{O(1)}$, each call to R takes $T(k)|x|^{O(1)}$ time, and the OR function is called on a tuple whose maximum parameter is bounded by the parameter bound of R. Hence the output parameter is bounded by a computable function of k.

If $(x, k) \in \Pi_{\mathrm{YES}}$, then $\Pr_r[\phi_{x,k}(r) = 1] \geq 2/3$. In the small case, the list contains a successful tape; in the large case, the hitting set hits the accepting set of $\phi_{x,k}$. Thus at least one input to C is a YES instance, and the output is YES. If $(x, k) \in \Pi_{\mathrm{NO}}$, then soundness of R says that every tape gives a NO instance of Π', and the OR function outputs NO. Therefore we have a deterministic FPT many-one reduction from Π to Π', and the claimed hardness follows from item (1). ◀

5 Conditional Hardness of MDP and SVP

We now construct the OR functions needed in Theorems 34 and 35 and apply the derandomization theorem to MDP and SVP.

5.1 OR functions

▶ **Lemma 36.** *Let $q \geq 2$ be a prime power, and let $\gamma \geq 1$ be a constant. Then γ-GapMDP$_q$, parameterized by the distance threshold, has an FPT OR function.*

Proof. Given a tuple of strings, discard all syntactically invalid strings and keep the valid instances $(\mathbf{G}_i, k_i)$. If none remain, output any fixed NO instance. Let $\kappa = \max_i k_i$, $K = \mathrm{lcm}(1, \ldots, \kappa)$, and $c_i = K/k_i$. Let $\widetilde{\mathbf{G}}_i$ be obtained from $\mathbf{G}_i$ by repeating each row c_i times, and output the block-diagonal generator matrix

$$\mathbf{G}_{\mathrm{out}} = \mathrm{diag}(\widetilde{\mathbf{G}}_1, \ldots, \widetilde{\mathbf{G}}_s) \quad \text{with threshold } K.$$

Repeating rows multiplies the minimum distance by c_i, and the minimum distance of a block-diagonal code is the minimum of the block minimum distances. Hence

$$\lambda(\mathcal{C}(\mathbf{G}_{\mathrm{out}})) = \min_i c_i \lambda(\mathcal{C}(\mathbf{G}_i)).$$

Thus a YES input block gives distance at most K, while if all retained inputs are NO then every block has distance larger than γK. The output threshold K depends only on κ, and the construction runs in time $T(\kappa)N^{O(1)}$. This is an FPT OR function. ◀

▶ **Lemma 37.** *Let $p \in [1, \infty)$ and let $\gamma \geq 1$ be a constant. Then γ-GapSVP$_p$ has a polynomial-time OR function.*

Proof. Keep the valid instances $(\mathbf{B}_i, d_i)$ with $d_i \geq 1$, and output a fixed NO instance if none remain. Set $D = \prod_i d_i$, $c_i = D/d_i$, and $\widetilde{\mathbf{B}}_i = c_i \mathbf{B}_i$. Output

$$\mathbf{B}_{\mathrm{out}} = \mathrm{diag}(\widetilde{\mathbf{B}}_1, \ldots, \widetilde{\mathbf{B}}_s) \quad \text{with threshold } D.$$

Scaling a basis by c_i scales $\lambda_1^{(p)}$ by c_i, and a block-diagonal lattice has shortest-vector length equal to the minimum over blocks. Therefore

$$\lambda_1^{(p)}(\mathcal{L}(\mathbf{B}_{\mathrm{out}})) = \min_i c_i \lambda_1^{(p)}(\mathcal{L}(\mathbf{B}_i)).$$

The OR guarantee follows exactly as above. The bit length of D is at most the sum of the bit lengths of the input thresholds, so the construction is polynomial time. ◀

▶ **Lemma 38.** *Let $p \in [1, \infty)$ and let $\gamma \geq 1$ be a constant. Then γ-GapSVP$_p$, parameterized by the threshold, has an FPT OR function.*

Proof. Use the same block-diagonal construction as in Lemma 37, but replace $D = \prod_i d_i$ by $D = \mathrm{lcm}(1, \ldots, \kappa)$, where $\kappa = \max_i d_i$. Then D depends only on the largest input parameter, and the same distance calculation proves the OR guarantee. The construction runs in time $T(\kappa)N^{O(1)}$, and the output parameter is D. ◀

5.2 Applications

We will use the following lemma to verify item (4) in Theorem 35.

▶ **Lemma 39.** *Let R be a randomized* FPT *many-one reduction from Π to Π'. Suppose that, after padding the random tape with ignored bits, its length μ on input (x, k) is at least the running-time bound of R on that input, and set $m = |\langle x, k\rangle| + \mu$. If $\Pi'_{\mathrm{YES}} \in$ NP, then R has SAT-circuitizable success predicates.*

Proof. For fixed (x, k), view the success predicate as a function of $r \in \{0,1\}^m$, where only the first μ bits are used as the random tape. Since the running time, and hence the output length, is at most $\mu \leq m$, compute $R((x, k); r)$ by a polynomial-size circuit. Then compose with a polynomial-time reduction from Π'_{YES} to SAT, and finally make one SAT-oracle query. ◀

Therefore, the desired claims follow.

Proof of Theorem 4. Fix q and γ. We apply Theorem 35 to the randomized reduction with one-sided error underlying Theorem 1; equivalently, the source is the W[1]-hard problem $4q$-GapNCP$_q$ from Theorem 13, followed by the constant number of tensorings needed to reach the target approximation factor. The target γ-GapMDP$_q$ has an FPT OR function by Lemma 36.

The YES language of γ-GapMDP$_q$ is in NP. The concrete reduction can be padded with ignored random bits so that the random tape length is at least its FPT running-time bound. Hence Lemma 39 gives SAT-circuitizable success predicates. All hypotheses of Theorem 35 hold, so γ-GapMDP$_q$ is W[1]-hard under deterministic FPT many-one reductions. ◀

Proof of Theorem 6. Fix $\gamma \geq 1$. By Theorem 15, there is a randomized polynomial-time reduction with one-sided error from an NP-hard language to γ-GapSVP$_2$. The YES language of γ-GapSVP$_2$ is in NP. By Lemma 37, γ-GapSVP$_2$ has a polynomial-time OR function. Therefore Theorem 34 applies and yields deterministic polynomial-time NP-hardness. ◀

Proof of Theorem 5. Fix an integer $p \geq 1$ and $\gamma < 2^{1/p}$. Choose a rational η with $\gamma < \eta < 2^{1/p}$. It is enough to prove deterministic W[1]-hardness of η-GapSVP$_p$, since the identity map reduces η-GapSVP$_p$ to γ-GapSVP$_p$ on promised instances.

By Theorem 2, η-GapSVP$_p$ is W[1]-hard under randomized FPT many-one reductions with one-sided error; specifically, the source is the W[1]-hard Γ-GapCVP$_p$ from Theorem 17, where Γ is the source approximation factor in Theorem 33. The target has an FPT OR function by Lemma 38.

Since p is fixed, the YES language of η-GapSVP$_p$ is in NP. The reduction in Theorem 33 can be padded with ignored random bits so that the random tape length is at least its FPT running-time bound. By Lemma 39, it has SAT-circuitizable success predicates. Applying Theorem 35 gives deterministic FPT hardness of η-GapSVP$_p$, and hence of γ-GapSVP$_p$. ◀

References

1 Divesh Aggarwal, Rishav Gupta, Aditya Morolia, and Chuanqi Zhang. Mind the Gap? Not for SVP Hardness under ETH! In *ICALP*, 2026.

2 Divesh Aggarwal and Noah Stephens-Davidowitz. (Gap/S)ETH hardness of SVP. In *STOC*, 2018.

3 E. Agrell, T. Eriksson, A. Vardy, and K. Zeger. Closest point search in lattices. *IEEE Transactions on Information Theory*, 48(8):2201–2214, 2002. doi:10.1109/TIT.2002.800499.

4 Miklós Ajtai. The shortest vector problem in L2 is NP-hard for randomized reductions (extended abstract). In *STOC*, 1998.

5 Per Austrin and Subhash Khot. A Simple Deterministic Reduction for the Gap Minimum Distance of Code Problem. *IEEE Transactions on Information Theory*, 60(10):6636–6645, 2014. Preliminary version in ICALP 2011. doi:10.1109/TIT.2014.2340869.

6 Marshall Ball, Elette Boyle, Akshay Degwekar, Apoorvaa Deshpande, Alon Rosen, Vinod Vaikuntanathan, and Prashant Nalini Vasudevan. Cryptography from Information Loss. In *ITCS*, 2020.

7 Huck Bennett, Mahdi Cheraghchi, Venkatesan Guruswami, and João Ribeiro. Parameterized Inapproximability of the Minimum Distance Problem over All Fields and the Shortest Vector Problem in All ℓ_p Norms. In *STOC*, 2023.

8 Huck Bennett and Chris Peikert. Hardness of Bounded Distance Decoding on Lattices in ℓ_p Norms. In *CCC*, 2020.

9 Huck Bennett and Chris Peikert. Hardness of the (Approximate) Shortest Vector Problem: A Simple Proof via Reed-Solomon Codes. In *RANDOM/APPROX*, 2023.

10 Huck Bennett, Chris Peikert, and Yi Tang. Improved hardness of BDD and SVP under Gap-(S)ETH. In *ITCS*, 2022.

11 Arnab Bhattacharyya, Édouard Bonnet, László Egri, Suprovat Ghoshal, Karthik C. S., Bingkai Lin, Pasin Manurangsi, and Dániel Marx. Parameterized Intractability of Even Set and Shortest Vector Problem. *Journal of the ACM*, 68(3), 2021. doi:10.1145/3444942.

12 Richard Chang and Jim Kadin. On Computing Boolean Connectives of Characteristic Functions. *Mathematical Systems Theory*, 28(3):173–198, 1995. doi:10.1007/BF01303054.

13 Qi Cheng and Daqing Wan. A Deterministic Reduction for the Gap Minimum Distance Problem. *IEEE Transactions on Information Theory*, 58(11):6935–6941, 2012. Preliminary version in STOC 2009. doi:10.1109/TIT.2012.2209198.

14 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

15 Rodney G. Downey and Michael R. Fellows. Fixed-Parameter Tractability and Completeness I: Basic Results. *SIAM Journal on Computing*, 24(4):873–921, 1995. doi:10.1137/S0097539792228228.

16 Rodney G. Downey and Michael R. Fellows. Fixed-Parameter Tractability and Completeness II: On Completeness for W[1]. *Theoretical Computer Science*, 141(1–2):109–131, 1995. doi:10.1016/0304-3975(94)00097-3.

17 Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999. doi:10.1007/978-1-4612-0515-9.

18 Ilya Dumer, Daniele Micciancio, and Madhu Sudan. Hardness of approximating the minimum distance of a linear code. *IEEE Transactions on Information Theory*, 49(1):22–37, 2003. Preliminary version in FOCS 1999. doi:10.1109/TIT.2002.806118.

19 Oded Goldreich. *On promise problems: A survey*, volume 3895 of *Lecture Notes in Computer Science*, pages 254–290. Springer, 2006. doi:10.1007/11685654_12.

20 Oded Goldreich and Shafi Goldwasser. On the Limits of Nonapproximability of Lattice Problems. *Journal of Computer and System Sciences*, 60(3):540–563, 2000. Preliminary version in STOC 1998. doi:10.1006/JCSS.1999.1686.

21 Isaac M Hair and Amit Sahai. Deterministic Hardness of Approximation For SVP in all Finite ℓ_p Norms, 2026. doi:10.48550/arXiv.2604.01451.

22 Isaac M. Hair and Amit Sahai. SVP_p Is Deterministically NP-Hard for All $p > 2$, Even to Approximate within a Factor of $2^{\log^{1-\varepsilon} n}$. In *STOC*, 2026.

23 Ishay Haviv and Oded Regev. Tensor-based Hardness of the Shortest Vector Problem to within Almost Polynomial Factors. *Theory of Computing*, 8(23):513–531, 2012. Preliminary version in STOC 2007. doi:10.4086/TOC.2012.V008A023.

24 Yahli Hecht and Muli Safra. Deterministic Hardness of Approximation of Unique-SVP and GapSVP in ℓ_p Norms for $p > 2$. In *STOC*, 2026.

25 Markus Hittmeir. Fine-grained deterministic hardness of the shortest vector problem, 2026. arXiv:2511.01626.

26 Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: derandomizing the XOR lemma. In *STOC*, 1997.

27 Subhash Khot. Hardness of approximating the shortest vector problem in lattices. *Journal of the ACM*, 52(5):789–808, 2005. Preliminary version in FOCS 2004. doi:10.1145/1089023.1089027.

28 Subhash Khot. Hardness of approximating the Shortest Vector Problem in high ℓ_p norms. *Journal of Computer and System Sciences*, 72(2):206–219, 2006. Preliminary version in FOCS 2003. doi:10.1016/J.JCSS.2005.07.002.

29 Adam R. Klivans and Dieter van Melkebeek. Graph Nonisomorphism Has Subexponential Size Proofs Unless the Polynomial-Time Hierarchy Collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002. Preliminary version in STOC 1999. doi:10.1137/S0097539700389652.

30 Shuangle Li, Bingkai Lin, and Yuwei Liu. Improved Lower Bounds for Approximating Parameterized Nearest Codeword and Related Problems Under ETH. In *ICALP*, 2024.

31 Shu Lin and Juane Li. *Fundamentals of Classical and Modern Error-Correcting Codes*. Cambridge University Press, 2021.

32 Daniele Micciancio. The Shortest Vector in a Lattice is Hard to Approximate to within Some Constant. *SIAM Journal on Computing*, 30(6):2008–2035, 2001. Preliminary version in FOCS 1998. doi:10.1137/S0097539700373039.

33 Daniele Micciancio. Inapproximability of the Shortest Vector Problem: Toward a Deterministic Reduction. *Theory of Computing*, 8(22):487–512, 2012. doi:10.4086/TOC.2012.V008A022.

34 Daniele Micciancio. Locally Dense Codes. In *CCC*, 2014.

35 Chris Peikert. A Decade of Lattice Cryptography. *Foundations and Trends® in Theoretical Computer Science*, 10(4):283–424, 2016. doi:10.1561/0400000074.

36 I. S. Reed and G. Solomon. Polynomial Codes Over Certain Finite Fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

37 Ronen Shaltiel and Jad Silbak. Explicit Codes for Poly-Size Circuits and Functions that are Hard to Sample on Low Entropy Distributions, 2023. Electronic Colloquium onComputational Complexity (ECCC). URL: https://eccc.weizmann.ac.il/report/2023/149/.

38 Ronen Shaltiel and Christopher Umans. Pseudorandomness for Approximate Counting and Sampling. *Computational Complexity*, 15(4):298–341, 2006. doi:10.1007/S00037-007-0218-9.

39 Peter van Emde Boas. Another NP-complete partition problem and the complexity of computing short vectors in a lattice. *Technical Report, Department of Mathematics, University of Amsterdam*, 1981.

40 Daqing Wan. NP-hardness of SVP in Euclidean Space, 2026. doi:10.48550/arXiv.2603.27398.

41 Violetta Weger, Niklas Gassner, and Joachim Rosenthal. A Survey on Code-Based Cryptography, 2024. arXiv:2201.07119.

Sequential Sweeps and High Dimensional Expansion

Vedat Levi Alev
Department of Mathematics, University of Haifa, Israel
Department of Computer Science, Ariel University, Israel

Ori Parzanchevski
Einstein Institute of Mathematics, Hebrew University of Jerusalem, Israel

Abstract

It is well known that the spectral gap of the down-up walk over an n-partite simplicial complex (also known as Glauber dynamics) cannot be better than $O(1/n)$ due to natural obstructions such as coboundaries. We study an alternative random walk over partite simplicial complexes known as the sequential sweep or the systematic scan Glauber dynamics: Whereas the down-up walk at each step selects a random coordinate and updates it based on the remaining coordinates, the sequential sweep goes through each of the coordinates one by one in a deterministic order and applies the same update operation. It is natural, thus, to compare n-steps of the down-up walk with a single step of the sequential sweep. Interestingly, while the spectral gap of the n-th power of the down-up walk is still bounded from above by a constant, under a strong enough local spectral assumption (in the sense of Gur, Lifschitz, Liu, STOC 2022) we can show that the spectral gap of this walk can be arbitrarily close to 1. We also study other isoperimetric inequalities for these walks, and show that under the assumptions of local entropy contraction (related to the considerations of Gur, Lifschitz, Liu), these walks satisfy an entropy contraction inequality. Concretely, we generalize a result of Lubetzky, Lubotzky, and Parzanchevski (Journal of the EMS) about the rapid mixing of sequential sweep in Ramanujan complexes to suitable high dimensional expanders.

2012 ACM Subject Classification Theory of computation → Random walks and Markov chains; Theory of computation → Expander graphs and randomness extractors

Keywords and phrases Random walks, high dimensional expanders, Ramanujan complexes, systematic scan, Glauber dynamics

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.47

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2312.02089

Funding *Vedat Levi Alev*: Supported by the ERC grant of Alex Lubotzky (European Union's Horizon 2020/882751), the ISF grant 2669/21 and ERC grant 834735 of Gil Kalai, ISF grant 2990/21 of Ori Parzanchevski, and ISF grant 721/2024 of Uriya First.
Ori Parzanchevski: Supported by the ISF grant 2990/21

Acknowledgements VLA would like to thank Max Hopkins for many insightful discussions.

1 Introduction

Let Ω be a finite set of n-tuples, i.e. $\Omega \subset U_1 \times U_2 \times \cdots \times U_n$, where each U_i is a finite set. Given a tuple $\omega \in \Omega$, for each $i \in [n]$, the i-th update operation $\mathsf{update}_i(\omega)$ returns a uniformly random tuple from Ω that is conditioned to agree with ω everywhere except for the i-th coordinate, i.e.

$$\Pr_{\omega' \sim \Omega}[\mathsf{update}_i(\omega) = \omega'] = \Pr_{\omega' \sim \Omega}\left[\omega' \,\middle|\, \omega_j = \omega'_j \ \text{ for all } \ j \in [n] \setminus i\right].$$

© Vedat Levi Alev and Ori Parzanchevski;
licensed under Creative Commons License CC-BY 4.0

The main object of study in this paper will be the random walk called sequential sweep $\mathsf{P}_{\mathtt{seq}}$ on the collection Ω, otherwise known as the systematic scan Glauber dynamics[1] on Ω: A single step of this dynamics moves between tuples by applying the n update operations in sequential fashion, i.e. writing $\omega^{(t)}$ for the tuple visited by this dynamics at step t, $\omega^{(t+1)}$ is the result of the following random update operations,

$$\omega^{(t+1)} = (\mathsf{update}_n \circ \mathsf{update}_{n-1} \circ \cdots \circ \mathsf{update}_1)(\omega^{(t)}) = \mathsf{update}_n(\mathsf{update}_{n-1}(\cdots(\mathsf{update}_1(\omega^{(t)})))).$$

It is easy to see that the stationary distribution for this random walk is the uniform distribution on Ω.

It is instructive to compare $\mathsf{P}_{\mathtt{seq}}$ with the the down-up walk $\mathsf{P}_{\mathtt{gd}}$ on Ω also known as the Glauber dynamics, which moves between tuples by first sampling a random coordinate $i \in [n]$ and then applying the operation $\mathsf{update}_i(\bullet)$. In other words, $\mathsf{P}_{\mathtt{seq}}$ can be thought as simulating n successive steps of $\mathsf{P}_{\mathtt{gd}}$ in which we condition the coordinate that is updated at the i-th step to be i for all $i \in [n]$.

1.1 HDX, Spectral Independence, and ε-products

We recall the notion of local-spectral expansion in (partite) simplicial complexes introduced in [40, 23, 42, 51, 22, 4]. A (finite) simplicial complex of rank n is a collection of subsets of size at most n of a finite set U, i.e. $X \subset \binom{U}{\leq n}$, which is closed under containment. X is called partite, if we can partition U into n disjoint subsets $U_1, \ldots, U_n$ such that any set from X of size n contains a unique element from each U_i – the sets U_i are called the sides of the simplicial complex. In particular, each $\omega \in X$ with $|\omega| = n$ can be treated as a tuple $(\omega_1, \ldots, \omega_n)$ where $\omega_i \in U_i$. We will write $X^{(i)} = \{\omega \in X : |\omega| = i\}$. We will also write $\omega_S = (\omega_s)_{s \in S}$ for all $\omega \in X^{(n)}$.

The local structures in X, are described by its links. For a face $\alpha \in X$, we describe the weighted graph $G^{(\alpha)} = (V_\alpha, E_\alpha, w_\alpha)$ as follows. We have,

$$V_\alpha = \{u \in U : \alpha \cup u \in X\} \text{ and } E_\alpha = \left\{uv \in \binom{U}{2} : \alpha \cup \{u,v\} \in X\right\}.$$

Every edge uv is assigned a weight based on the number of subsets in X of size n that contain $\alpha \cup \{u, v\}$, i.e.

$$w_\alpha(uv) = |\{\omega \in X : |\omega| = n, \omega \supset \alpha \cup \{u,v\}\}|.$$

Writing $G^{(\alpha)}$ for the random walk matrix of $G^{(\alpha)}$ also, we observe that $\lambda_2(G^{(\alpha)})$ measures an amortized notion of the correlation between the vertices in V_α. Formally, we say that the simplicial complex X is a $\vec{\gamma} = (\gamma_0, \ldots, \gamma_{n-2})$-local spectral expander if for each $\alpha \in X$ with $|\alpha| \leq n-2$, we have $\lambda_2(G^{(\alpha)}) \leq \gamma_{|\alpha|}$. When $n = 2$, this notion is equivalent to the usual notion of a spectral expander, [37, 44].

For our purposes an alternative but related notion of high-dimensional expansion introduced in [34, 21] will be more useful to consider. Whereas, we have said that $\lambda_2(G^{(\alpha)})$ measures an amortized notion of the correlation between vertices in V_α, we will be more interested in measuring a worst case notion of the correlation between the vertices in V_α which come from pre-specified parts U_i and U_j. Formally, we are interested in the second eigenvalue of the following bipartite graph $G_{i,j}^{(\alpha)} = (V_{ij}^{(\alpha)}, E_{ij}^{(\alpha)}, w_{ij}^{(\alpha)})$ for each $\alpha \in X$ such

[1] More precisely, n steps of the systematic scan dynamics is the walk we call sequential sweep here.

that $|\alpha \cap U_i| = |\alpha \cap U_j| = 0$. In words, α can be thought as an assignment to a subset of coordinates $S \subset [n]$, such that $\{i,j\} \notin S$. Here,

$$V_{ij}^{(\alpha)} = \{u \in U_i \cup U_j : \alpha \cup u \in X\} \quad \text{and} \quad E_{ij}^{(\alpha)} = \{uv : u \in U_i, v \in U_j, \alpha \cup \{u,v\} \in X\}.$$

Similarly, we assign each edge uv a weight between the number of subsets in X of size n containing $\alpha \cup \{u,v\}$, i.e.

$$w_{ij}^{(\alpha)}(uv) = \left|\left\{\omega \in X^{(n)} : |\omega| = n, \omega_S = \alpha, \omega_i = u, \omega_j = v\right\}\right|,$$

where we have assumed that α to be a tuple with coordinates in $S \subset [n]$, $u \in U_i$, and $v \in V_j$. We will call the simplicial complex $\underline{\vec{\varepsilon} = (\varepsilon_0, \ldots, \varepsilon_{n-2})\text{-product}}$ if for all α and i,j we have $\lambda_2(G_{i,j}^{(\alpha)}) \le \varepsilon_{|\alpha|}$. We observe, that the quantity $\lambda_2(G_{i,j}^{(\alpha)}) \le \varepsilon_{|\alpha|}$ quantifies the correlation between the i-th and j-th coordinate on an n-tuple drawn from X conditioned on agreeing with α. We say X is completely ε-product if it is $\vec{\varepsilon} = (\varepsilon_i)_{i=0}^{n-2}$-product where $\varepsilon_i \le \varepsilon$ for all $i = 0, \ldots, n-2$.

In [34] this definition was used in proving hypercontractive estimates for the noise operator over simplicial complexes and in [21], it was used in the context of agreement testing. We recall, that every strong enough local partite expander is also $\vec{\varepsilon}$-product [21, Corollary 7.6]:

▶ **Theorem 1.** *Let X be an n-partite simplicial complex. If X is a $\vec{\gamma}$-local spectral expander with $\gamma_{n-2} \le \frac{\varepsilon}{(n-2)\varepsilon+1}$, then X is also completely ε-product.*

And also, recall the following rudimentary remark

▶ Remark 2. [2] Let X be an n-partite simplicial complex. If X is a $\vec{\gamma}$-local spectral expander with $\gamma_i \le \frac{\varepsilon_i}{n-i-1}$, then X is also $(\varepsilon_0, \ldots, \varepsilon_{n-2})$-product.

More generally, we will consider the following bipartite graphs $G_{I,J}^{(\alpha)} = (V_{I,J}^{(\alpha)}, E_{I,J}^{(\alpha)}, w_{I,J}^{(\alpha)})$ given $\alpha \in X$, and disjoint sets I and J such that $|\alpha \cap U_k| = 0$ for all $k \in I \cup J$. Similarly, we are thinking of α as a partial assignment to a subset of the coordinates of an n-tuple which do not assign anything to the coordinates in I or J. Here,

$$V_{I,J} = \left\{\omega_I \in \prod_{i \in I} U_i : \alpha \cup \omega_I \in X\right\} \cup \left\{\omega_J \in \prod_{j \in J} U_j : \alpha \cup \omega_J \in X\right\},$$

and

$$E_{I,J} = \left\{\omega_I \omega_J : \omega_I \in \prod_{i \in I}, \omega_J \in \prod_{j \in I}, \alpha \cup \omega_I \cup \omega_J \in X\right\},$$

and assign a weight on each edge $\omega_I \omega_J \in E_{I,J}$ based on the number of subsets in X of size n that contain $\alpha \cup \{u,v\}$, i.e.

$$w_{I,J}^{(\alpha)}(\omega_I \omega_J) = \left|\left\{\bar{\omega} \in X^{(n)} : \bar{\omega}_S = \alpha, \bar{\omega}_I = \omega_I, \bar{\omega}_S = \omega_S\right\}\right|.$$

where we have assumed α to be a partial assignment to the coordinates in S. Similarly to before, $\lambda_2(G_{I,J}^{(\alpha)})$ is a measure of correlation between the coordinates on I and the coordinates in J.

In [21, 34], the following bound is shown,

▶ **Theorem 3.** *If X is an n-partite simplicial complex with parts $U_1, \ldots, U_n$ which is $\vec{\varepsilon}$-product (where $\varepsilon_i \le \varepsilon$ for all $i = 0, \ldots, n-2$), then for all α satisfying $|\alpha \cap U_k| = 0$ for $k \in I \cup J$, we have $\lambda_2(G_{I,J}^{(\alpha)})^2 \le |I||J| \cdot \varepsilon^2$.*

[2] We will prove this observation formally.

1.2 Results and Techniques

Let X be an n-partite simplicial complex. We write,

$$\varepsilon^{I\to J} := \varepsilon_X^{I\to J} = \max\Big\{\lambda_2(G_{I,J}^{(\alpha)}) : \alpha \text{ is an assignment to the coordinates in } [n]\setminus(I\cup J)\Big\}.$$

In particular, $\varepsilon^{I\to J}$ quantifies the maximum correlation between the coordinates in I and J from an n-tuple drawn from X given any assignment α to the coordinates in $(I\cup J)^c$.

We have,

▶ **Theorem 4** (Informal Version of Theorem 25). *Let X be an n-partite simplicial complex. We have,*

$$\sigma_2(\mathsf{P}_{\mathsf{seq}})^2 \le 1 - \prod_{j=2}^{n}(1-(\varepsilon^{[j-1]\to j})^2),$$

where $\sigma_2(\bullet)$ denotes the second largest singular value of $\bullet$.

In particular, writing $\mathsf{gap}(\bullet) = 1-\sigma_2(\bullet)$, we have

$$\mathsf{gap}(\mathsf{P}_{\mathsf{seq}}) \ge 1 - \sqrt{1-\prod_{j=2}^{n}(1-(\varepsilon^{[j-1]\to j})^2)}.$$

We also give an improved version of Theorem 3,

▶ **Theorem 5** (Simplified Version of Theorem 30). *Let (X,π) be an n-partite simplicial complex where π is $\vec{\varepsilon}$-product. For any assignment α to $S\subset[n]$ such that $S\cap[j]=\varnothing$, we have,*

$$\sigma_2(G^{(\alpha)}_{\{j\},[j-1]})^2 \le 1 - \prod_{q=0}^{j-2}(1-\varepsilon^2_{|\alpha|+q})$$

In particular, we have

$$(\varepsilon^{j\to[j-1]})^2 \le 1 - \prod_{q=0}^{j-1}(1-\varepsilon^2_{n-j+q})$$

In the full version of our paper, we study the expansion parameters $\varepsilon^{j\to[j-1]}$ in Ramanujan complexes of thickness p. We summarize our results for these complexes below in the following theorem.[3]

▶ **Theorem 6.** *Let X be a Ramanujan complex of thickness p and dimension $(d-1)$. We have,*

$$\Big(\varepsilon^{[0,j-1]\to j}\Big)^2 = \frac{1}{p+1}\big(1-\Theta\big(p^{-j}\big)\big) \quad \text{and} \quad \Big(\varepsilon^{[0,\ldots,d-2]\to d-1}\Big)^2 = \frac{1}{p+1}\cdot\big(1-\Theta\big(p^{-j}\big)\big) \tag{1}$$

for all $j=1,\ldots,d-2$. Consequently,

$$\sigma_2\big(\mathsf{P}_{\mathsf{seq}}\big)^2 \le \frac{d+1}{p} - O\left(\frac{d^2}{p^2}\right)$$

where N is the number of vertices in X.

[3] Recall that a Ramanujan complex of dimension $d-1$ is a d-partite simplicial complex where the colors are named from 0 to $d-1$.

We note that the expansion parameters in Equation (1) are computed exactly and they do not suffer the co-dimension dependent blowups from Theorem 5, which would replace the RHS of the expansion parameters in Equation (1) with *roughly* $(j-1)/p$ and $(d-1)/(p+1)$, compare with Theorem 3. While our techniques fall short of recovering the optimal mixing results of [46], we hope our techniques will find further applications in the study of *spectral* expansion of Ramanujan complexes. We note that our spectral bound does not follow from the spectral techniques of [46] as this work concerns itself with spectral radii of non-backtracking operators, which in our context do not neatly translate into a bound on the singular values.

We note that Theorem 4 is proven using geometric considerations. For each $I \subset [n]$, we define the subspace $\mathcal{U}_I \subset \mathbb{R}^{X^{(n)}}$ as

$$\mathcal{U}_I = \operatorname{span}\Big\{ \boldsymbol{f} \in \mathbb{R}^{X^{(n)}} : \boldsymbol{f}(\omega) = \boldsymbol{f}(\bar{\omega}) \ \text{ for all } \ \omega, \bar{\omega} \in X^{(n)} \text{ such that } \omega_{[n]\setminus I} = \bar{\omega}_{[n]\setminus I} \Big\}$$

and we will write $\mathsf{Q}_I = \mathrm{Proj}(\mathcal{U}_I)$ for the orthogonal projection to $\mathcal{U}_I$. It is well known that $\mathsf{P}_{\mathtt{seq}} = \mathsf{Q}_{\{1\}} \cdots \mathsf{Q}_{\{n\}}$, e.g. see [6]. We recall that the cosine between two subspaces $\mathcal{U}, \mathcal{V} \in \mathbb{R}^{X^{(n)}}$ is defined as,

$$\cos(\mathcal{U}, \mathcal{V}) = \max\big\{ \langle \boldsymbol{u}, \boldsymbol{v} \rangle_\pi : \boldsymbol{u} \in \mathcal{U} \cap (\mathcal{U} \cap \mathcal{V})^\perp, \boldsymbol{v} \in (\mathcal{U} \cap \mathcal{V})^\perp, \|\boldsymbol{u}\|_\pi = \|\boldsymbol{v}\|_\pi = 1 \big\},$$

Similarly, $\sin^2(\mathcal{U}, \mathcal{V}) = 1 - \cos^2(\mathcal{U}, \mathcal{V})$. Now, we can follow in the footsteps of [6], who has proven estimates on the spectral gap of the sequential sweep in the Gaussian setting, by appealing to the following theorem:

▶ **Theorem 7** ([59]). *Let $\mathsf{Q}_1, \ldots, \mathsf{Q}_p \in \mathbb{R}^{\Omega \times \Omega}$ be projection operators to sub-spaces $\mathcal{U}_1, \ldots, \mathcal{U}_p$. Set $\mathcal{V}_j = \bigcap_{i=1}^{j} \mathcal{U}_i$. Writing $\mathsf{Q}_\star$ for the projection to $\mathcal{V}_p$, we have*

$$\|\mathsf{Q}_1 \cdots \mathsf{Q}_p \boldsymbol{f} - \mathsf{Q}_\star \boldsymbol{f}\|^2 \le \left(1 - \prod_{j=2}^{p} \sin^2(\mathcal{U}_j, \mathcal{V}_{j-1}) \right) \cdot \|\boldsymbol{f} - \mathsf{Q}_\star \boldsymbol{f}\|^2.$$

Now, Theorem 4 follows from the following geometric interpretation of $\varepsilon^{I \to J}$:

▶ **Theorem 8** (Informal Version of Theorem 29 and Proposition 27). *Let X be an n-partite simplicial complex $I, J \subset [n]$ such that $I \cap J = \varnothing$. Under suitable assumptions,*

$$\cos(\mathcal{U}_I, \mathcal{U}_J) \le \varepsilon^{I \to J} \quad \text{and} \quad \bigcap_{i \in I} \mathcal{U}_i = \mathcal{U}_I.$$

We also show an analog of Theorem 4 for contraction in relative entropy: We recall that the KL-divergence between the distributions μ and π is defined as $\mathcal{D}(\mu \parallel \pi) = \mathbb{E}_{x \sim \pi} \left[\frac{\mu(x)}{\pi(x)} \log \frac{\mu(x)}{\pi(x)} \right]$. The entropy contraction factor $\mathsf{ec}(\mathsf{P})$ of the random walk operator $\mathsf{P} \in \mathbb{R}^{V \times V}$ with stationary measure π is defined as,

$$\mathsf{ec}(\mathsf{P}) = 1 - \kappa(\mathsf{P}) \ \text{ where } \ \kappa(\mathsf{P}) = \sup\left\{ \frac{\mathcal{D}(\mu \mathsf{P} \parallel \pi)}{\mathcal{D}(\mu \parallel \pi)} : \mu \text{ is a probability distribution on } V \right\}.$$

We define the following entropy contraction parameters,

$$\eta^{I \to J} = 1 - \kappa^{I \to J} \ \text{ where } \ \kappa^{I \to J} = \sup\left\{ \frac{\mathcal{D}(\mu \cdot G^{(\alpha)}_{I,J} \parallel \pi^{(\alpha)}_J)}{\mathcal{D}(\mu \parallel \pi^{(\alpha)}_I)} : \alpha \in X[(I \cup J)^c] \text{ and } \mu \in \triangle_{X_\alpha[I]} \right\}$$

where we have overloaded the notation $G^{(\alpha)}_{I,J}$ to mean the random walk from the part of $V^{(\alpha)}_{I,J}$ that describes the transition from assignments of I to assignments of J, and $\pi^{(\alpha)}_I$ and $\pi^{(\alpha)}_J$ are uniform distributions the assignments of I and assignments of J respectively.

In the full version of our paper, we prove

▶ **Theorem 9.** *Let X be an n-partite simplicial complex. Then,*

$$\mathsf{ec}(\mathsf{P}_{\mathsf{seq}}) \geq \prod_{j=1}^{n-1} \eta^{[j+1,n]\to j}. \tag{2}$$

In particular, writing π for the uniform distribution on $X^{(n)}$ we have

$$\mathcal{D}(\mu \mathsf{P}_{\mathsf{seq}} \parallel \pi) \leq (1 - \mathsf{ec}(\mathsf{P}_{\mathsf{seq}})) \cdot \mathcal{D}(\mu \parallel \pi).$$

The proof of Theorem 9 follows from an inductive argument. Our key observation is that conditional on the 1-st coordinate equalling any given value x, the distribution of $\mu\mathsf{P}_{\mathsf{seq}}$ on the remaining coordinates in $[2,\ldots,n]$ can be thought as the distribution resulting from a sequential sweep in a smaller complex X_x with faces $(\omega_2,\ldots,\omega_n)$ such that $(x,\omega_2,\ldots,\omega_n)$ is a facet of X. The proof is then completed using the well-known Garland method [30] and the chain-rule for KL divergences. There is one interesting *twist*: usually the Garland method considers the localizations of functions, which need not be normalized. Applying the Garland method to distributions, requires understanding how the *localized/pinned* distributions interplay with the graphs $G^{(\alpha)}_{I,J}$. To the best of our knowledge, this is where our proofs differ from existing *local-to-global* arguments of [4, 17, 33, 5]. We hope our proof techniques will find more applications in the future.

We note that both $\mathsf{ec}(\mathsf{P}_{\mathsf{seq}})$ and $\mathsf{gap}(\mathsf{P}_{\mathsf{seq}})$ are useful tools for bounding the mixing time $T(\mathsf{P}_{\mathsf{seq}})$ of the random walk $\mathsf{P}_{\mathsf{seq}}$, i.e. the number of steps one needs to take to come within small, say 0.01, distance of the uniform distribution. We have,

$$T(\mathsf{P}_{\mathsf{seq}}) \lesssim \left\lceil \frac{\log\big(|X^{(n)}|\big)}{\log\big(1/\sigma_2(\mathsf{P}_{\mathsf{seq}})\big)} \right\rceil \lesssim \left\lceil \frac{\log(|X^{(n)}|)}{\mathsf{gap}(\mathsf{P}_{\mathsf{seq}})} \right\rceil,$$

$$T(\mathsf{P}_{\mathsf{seq}}) \lesssim \left\lceil \frac{\log\log(|X^{(n)}|)}{\log(1/(1-\mathsf{ec}(\mathsf{P}_{\mathsf{seq}})))} \right\rceil \lesssim \left\lceil \frac{\log\log\big(|X^{(n)}|\big)}{\mathsf{ec}(\mathsf{P}_{\mathsf{seq}})} \right\rceil$$

When $\mathsf{gap}(\mathsf{P}_{\mathsf{seq}})$ and $\mathsf{ec}(\mathsf{P}_{\mathsf{seq}})$ tend to 1, the first bound in each inequality is significantly sharper than the second. However, as they get closer and closer to 0, the latter inequalities are essentially lossless. In particular, when $\mathsf{gap}(\mathsf{P}_{\mathsf{seq}})$ and $\mathsf{ec}(\mathsf{P}_{\mathsf{seq}})$ are of the same order, the bounds one obtains through $\mathsf{ec}(\mathsf{P}_{\mathsf{seq}})$ are typically much sharper, though typically this constant is much more troublesome to bound.

▶ Remark 10. We note that our proof technique for Theorem 9 is general enough to prove contraction estimates for $\mathsf{P}_{\mathsf{seq}}$ with respect to general Φ-divergences, $\mathcal{D}_\Phi(\mu \parallel \pi) = \mathbb{E}_{\omega\sim\pi}\left[\Phi\left(\frac{\mu(\omega)}{\pi(\omega)}\right)\right]$ for convex Φ, assuming that $G^{(\alpha)}_{[j+1,n],\{j\}}$ contract Φ-divergences. In particular, we could give an alternative proof for Theorem 4 by setting $\Phi(t) = (t-1)^2$ and adapting the proof of Theorem 9.[4] We will refrain from taking this route and carrying this out explicitly, as we believe that proving Theorem 4 through Theorem 7 better highlights the geometric nature of this result. We refer to [16, 14, 56] for more information on Φ-divergences and the closely related Φ-entropies, and inequalities for Φ-divergence contraction called *generalized* data-processing inequalities.

[4] This would correspond to the so-called χ^2-divergence. The bound by which a random walk contracts the χ^2 divergence is well understood to correspond to the (square of the) second singular value.

1.3 Related Work

A similar proof strategy in proving a spectral gap bound for the sequential sweep in the Gaussian setting was employed by [6] – there he shows that the cosine of the angle between subspaces can be bounded in terms of the covariance matrix of the underlying Gaussian distribution, which is where our analysis diverges from his. We rely on the Garland method [30] to bound these values, which has been a staple in high-dimensional expansion literature, e.g. [51, 52, 42, 22, 4, 53, 54]. In particular, there is a non-trivial overlap between our techniques and the ones employed in [49, 50, 53, 54, 31]. Similar considerations about angles between subspaces have also been investigated in the context of Kazhdan constants and the property (T), e.g. [43, 27, 39].

Our work is close in spirit to [46] which study a non-stuttering version of the sequential sweep $\mathsf{P}_{\mathtt{seq}}$ in Ramanujan complexes, called the geodesic flow. The main result of [46] is that for these geodesic flows on Ramanujan complexes, the total-variation mixing time is as small as possible up to an additive logarithmic constant. Namely, a k-regular flow on n states achieves ϵ-mixing after only $\log_k(n) + O(\log\log(n))$ steps. In fact, this result holds for a larger class of walks on so-called *Ramanujan digraphs* [55]. While we cannot recover this result, our results hold in much greater generality without assumption of Ramanujanness, and also show that in suitable high-dimensional expanders the sequential sweep improves upon down-up walks.

As already mentioned a significant amount of work has been done in investigating the mixing properties of the down-up walk $\mathsf{P}_{\mathtt{gd}}$ from considerations of high-dimensional expansion. See for example, [40, 23, 42, 22, 4] for works investigating local-spectral expansion (and variants) to get a bound on $\mathtt{gap}(\mathsf{P}_{\mathtt{gd}})$. Most of the concrete applications of these results rely on the spectral independence framework of [9, 28, 17] – we refer to the excellent lecture notes [60] and the references therein for more on this subject. Similarly, the works of [19, 33, 18, 7, 8] explore bounding $\mathtt{ec}(\mathsf{P}_{\mathtt{gd}})$ based on considerations of local entropy contraction. Our bound on $\mathtt{ec}(\mathsf{P}_{\mathtt{seq}})$ is proven using similar ideas. There has also been some work in connecting these concepts to classical Markov Chain techniques such as path coupling [1, 15], see e.g. [45, 12].

The random walks $G^{(\alpha)}_{I,J}$ that we have considered have also been studied in the works [21, 34], with applications in hypercontractivity and agreement testing. They are a natural generalization of swap-walks which have been studied in [3, 21]. Their study have found applications in approximation algorithms for constraint satisfaction problems [3], agreement testing [21], coding theory [2, 38], and hypercontractivity [10, 11, 34].

Although the sequential sweep $\mathsf{P}_{\mathtt{seq}}$ is a very natural random sampling algorithm considerably less is known about it and the results apply only in restricted and specialized settings, e.g. [20, 25, 35, 26]. These results show mixing by considering what is known as the Dobrushin matrix [24] whose entries give pessimistic estimates on the worst case dependencies between the coordinates. In [36, 57] the impact of the order of the vertices for $\mathsf{P}_{\mathtt{seq}}$ was studied and shown that the mixing time of the walk crucially depends on the order of the vertices. The relation between the relaxation time of $\mathsf{P}_{\mathtt{seq}}$ and $\mathsf{P}_{\mathtt{gd}}$ (with a given order) was also subject of investigation in [32]. We refer the readers to the papers mentioned here and the references therein for more on the topic of scan order and mixing time.

The sequence of updated coordinates in $\mathsf{P}_{\mathtt{seq}}$ can be thought as the set of vertices visited by a (deterministic) walk on the directed cyle $C_n = ([n], E)$ initiated from the vertex 1. It is now a natural idea to replace the directed circle with an arbitrary graph $G = ([n], E)$ and picking the updated vertices by means of a random walk on $[n]$. [5] shows that if G is a two-sided spectral expander, the mixing time of this new walk is comparable to the mixing time of $\mathsf{P}_{\mathtt{gd}}$ under mild assumptions. Their techniques fall short of establishing a mixing

time for the sequential sweep in blackbox fashion since the directed cycle has no two-sided expansion. However, we observe that their result essentially translates an $O(n \log n)$ mixing time for $\mathsf{P}_{\mathtt{gd}}$ into an $O(\frac{n \log n}{1-\lambda(G)})$ mixing time for this new-walk. Observing that this mixing time is roughly the cover time[5] of the graph G, suggesting that a tighter analysis could perhaps establish an $O(1)$-mixing time[6] for our sweep[7].

Finally, we mention the work [29], which proves worst case comparison theorems between the sequential sweep $\mathsf{P}_{\mathtt{seq}}$ and the Glauber dynamics $\mathsf{P}_{\mathtt{gd}}$, which shows that $\mathtt{gap}(\mathsf{P}_{\mathtt{seq}}) = \Omega(n^{-1}) \cdot \mathtt{gap}(\mathsf{P}_{\mathtt{gd}})$. They further give examples of complexes in which this bound is tight. We note that our focus in this work are complexes where $\mathtt{gap}(\mathsf{P}_{\mathtt{seq}})$ is significantly better than $\mathtt{gap}(\mathsf{P}_{\mathtt{gd}})$, thus our result can be thought as complementary to their result.

2 Preliminaries

2.1 Linear Algebra

Vectors and Inner Products

Given $\boldsymbol{f}, \boldsymbol{g} \in \mathbb{R}^V$, and a measure π on V such that $\pi(x) > 0$ for all $x \in V$, we use the notations $\langle \boldsymbol{f}, \boldsymbol{g} \rangle_\pi$ and $\|\boldsymbol{f}\|_\pi$ to denote the inner-product and the norm with respect to the distribution π, i.e.

$$\langle \boldsymbol{f}, \boldsymbol{g} \rangle_\pi = \mathop{\mathbb{E}}_{x \sim \pi} \boldsymbol{f}(x)\boldsymbol{g}(x) = \sum_{x \in V} \pi(x) \cdot \boldsymbol{f}(x)\boldsymbol{g}(x) \quad \text{and} \quad \|\boldsymbol{f}\|_\pi^2 = \langle \boldsymbol{f}, \boldsymbol{f} \rangle_\pi. \tag{3}$$

Matrices and Eigenvalues

In this section, we will recall some results concerning eigenvalues and eigenvectors of matrices.

We will call a matrix $\mathsf{B} \in \mathbb{R}^{U \times V}$ row-stochastic if all entries of B are non-negative and all rows of B sum up to 1, i.e.

$$\text{for all } u \in U, v \in V \quad \mathsf{B}(u, v) \geq 0 \quad \text{and} \quad \mathsf{B}\mathbf{1} = \mathbf{1}.$$

The adjoint of the operator $\mathsf{B} \in \mathbb{R}^{U \times V}$, with respect to the inner-products defined by the measures π_U and π_V on U and V, is the operator $\mathsf{B}^* \in \mathbb{R}^{V \times U}$ satisfying

$$\langle \boldsymbol{f}, \mathsf{B}\boldsymbol{g} \rangle_{\pi_U} = \langle \mathsf{B}^* \boldsymbol{f}, \boldsymbol{g} \rangle_{\pi_V} \quad \text{for all } \boldsymbol{f} \in \mathbb{R}^U, \boldsymbol{g} \in \mathbb{R}^V.$$

If $U = V$ and $\pi_U = \pi_V$, the operator B is called self-adjoint when $\mathsf{B}^* = \mathsf{B}$. If B is a row-stochastic matrix, we will call B^* the time-reversal of B with respect to π_U, π_V and say that B is reversible if $\mathsf{B} = \mathsf{B}^*$. It is well known that the operator $\mathsf{B}^* \in \mathbb{R}^{V \times U}$ is uniquely determined by the choice of $\mathsf{B} \in \mathbb{R}^{U \times V}$ and the inner-products defined by π_U and π_V (see e.g. [58, p. 318]),

▶ **Proposition 11.** *Let $\mathsf{B} \in \mathbb{R}^{U \times V}$ be arbitrary. We write B^* for the adjoint operator to B with respect to the inner-products defined by the distributions π_U and π_V. Then,*

$$\mathsf{B}^*(y, x) = \mathsf{B}(x, y) \cdot \frac{\pi_U(x)}{\pi_V(y)} \quad \textit{for all} \quad x \in U, y \in V.$$

[5] The number of steps following which the random is expected to have visited all the vertices after being initialized from an arbitrary vertex.
[6] i.e. $O(n)$ update steps
[7] where we observe that the cover time of the directed cycle is n, which would correspond to a single sweep

We also recall the following standard fact which is an immediate consequence of Proposition 11,

▶ **Proposition 12.** *If* $\mathsf{B} \in \mathbb{R}^{U,V}$ *is a row-stochastic matrix satisfying* $\pi_U \mathsf{B} = \pi_V$*, then the adjoint matrix* B^* *with respect to* π_U, π_V *is also row-stochastic and satisfies* $\pi_V \mathsf{B}^* = \pi_U$*.*

Throughout this paper, we will often deal with operators which are not self-adjoint, and therefore we will need to deal with singular values. Given an operator $\mathsf{B} \in \mathbb{R}^{V \times U}$ we will write $\sigma_i(\mathsf{B})$ for the i-th largest singular value of B, i.e. we have $\sigma_i(\mathsf{B}) = \sqrt{\lambda_i(\mathsf{B}^*\mathsf{B})}$.[8] This expression is well-defined since $\mathsf{B}^*\mathsf{B}$ is positive-semi definite and therefore $\lambda_i(\mathsf{B}^*\mathsf{B}) \geq 0$. We recall the following facts,

▶ **Proposition 13.** *Let* $\mathsf{B} \in \mathbb{R}^{U \times V}$ *be a row-stochastic matrix satisfying* $\pi_U \mathsf{B} = \pi_V$*. Then,*

1. $\sigma_1(\mathsf{B}) = 1$,
2. $\sigma_2(\mathsf{B}) =$
   $\max\left\{\langle \boldsymbol{f}, \mathsf{B}\boldsymbol{g}\rangle_{\pi_U} : \boldsymbol{f} \in \mathbb{R}^U, \boldsymbol{g} \in \mathbb{R}^V, \|\boldsymbol{f}\|_{\pi_U} = \|\boldsymbol{g}\|_{\pi_V} = 1, \langle \boldsymbol{f}, \mathbf{1}_U\rangle_{\pi_U} = \langle \boldsymbol{g}, \mathbf{1}_V\rangle_{\pi_V} = 0\right\}$.
3. $\sigma_2(\mathsf{B}) = \max\left\{\frac{\|(\mathsf{B}-\mathbf{1}_U\pi_V)\boldsymbol{f}\|_{\pi_U}}{\|\boldsymbol{f}-\mathbf{1}_V\pi_V\boldsymbol{f}\|_{\pi_V}} : \boldsymbol{f} \in \mathbb{R}^V, \|\boldsymbol{f} - \mathbf{1}_V\pi_V\boldsymbol{f}\|_{\pi_V} \neq 0\right\}$

A matrix $\mathsf{Q} \in \mathbb{R}^{V \times V}$ is called an orthogonal projection with respect to π if it is self adjoint with respect to π, i.e. $\mathsf{Q} = \mathsf{Q}^*$ and it satisfies the equation $\mathsf{Q}^2 = \mathsf{Q}$. We will say that Q is the orthogonal projection operator to a subspace $\mathcal{U} \subset \mathbb{R}^V$ if the image $\mathrm{im}(\mathsf{Q}) = \mathsf{Q}\mathbb{R}^V$ equals $\mathcal{U}$.

Finally, we will recall some results concerning the angles between subspaces and products of projection matrices. Let $\mathcal{U}, \mathcal{V} \subset \mathbb{R}^\Omega$ equipped with the inner-product π. The cosine of the angle between $\mathcal{U}$ and $\mathcal{V}$ is defined to be,

$$\cos(\mathcal{U}, \mathcal{V}) = \max\left\{\langle \boldsymbol{u}, \boldsymbol{v}\rangle_\pi : \boldsymbol{u} \in \mathcal{U} \cap (\mathcal{U} \cap \mathcal{V})^\perp, \boldsymbol{v} \in (\mathcal{U} \cap \mathcal{V})^\perp, \|\boldsymbol{u}\|_\pi = \|\boldsymbol{v}\|_\pi = 1\right\}, \tag{4}$$

▶ **Theorem 14** ([59])**.** *Let* $\mathsf{Q}_1, \ldots, \mathsf{Q}_p \in \mathbb{R}^{\Omega \times \Omega}$ *be projection operators to sub-spaces* $\mathcal{U}_1, \ldots, \mathcal{U}_p$*. Set* $\mathcal{V}_j = \bigcap_{i=1}^j \mathcal{U}_i$*. Writing* $\mathsf{Q}_\star$ *for the projection to* $\mathcal{V}_p$*, we have*

$$\|\mathsf{Q}_1 \cdots \mathsf{Q}_p \boldsymbol{f} - \mathsf{Q}_\star \boldsymbol{f}\|_\pi^2 \leq \left(1 - \prod_{j=2}^{p} \sin^2(\mathcal{U}_j, \mathcal{V}_{j-1})\right) \cdot \|\boldsymbol{f} - \mathsf{Q}_\star \boldsymbol{f}\|_\pi^2$$

where $\sin^2(\mathcal{U}, \mathcal{V}) = 1 - \cos^2(\mathcal{U}, \mathcal{V})$*.*

2.2 Probability Distributions and Relative Entropy

Let Ω be a finite set. We will write $\triangle_\Omega$ for the probability simplex with vertices Ω, i.e.

$$\triangle_\Omega := \left\{\mu : \Omega \to \mathbb{R}_{\geq 0} : \sum_{\omega \in \Omega} \mu(\omega) = 1\right\}.$$

We will adopt the convention of treating elements of $\mu \in \triangle_\Omega$ as row-vectors. For $\mu \in \triangle_\Omega$, we will write $\mathrm{supp}(\mu) = \{\omega \in \Omega : \mu(\omega) > 0\}$ for the support of μ. Throughout the paper, Ω (or X) will always be a set of n-tuples for some $n \geq 1$, i.e. there exists a finite sequence of sets $\Omega_1, \ldots, \Omega_n$ such that $\Omega \subset \Omega_1 \times \cdots \times \Omega_n$. Given a set $S \subset [n]$, we will write $\Omega[S]$ for the projection of Ω to the coordinates in S, i.e.

[8] Crucially, the i-th singular value always depends on the choice of the measures which define the adjoint matrix. We will mostly deal with row-stochastic matrices that satisfy $\pi_U \mathsf{B} = \pi_V$ for some choice of distributions π_U, π_V. The adjoint matrix B^* is then defined with respect to these distributions.

$$\Omega[S] = \{(\omega_s)_{s\in S} : (\omega_1, \ldots, \omega_n) \in \Omega\}.$$

The S-marginal distribution of $\mu \in \Omega$ is defined by,

$$\mu_S(\omega_S) = \sum_{\bar{\omega}\in\Omega[S^c]} \mu(\omega_S \oplus \bar{\omega}) \quad \text{for all} \quad \omega_S \in \Omega[S], \tag{5}$$

where $S^c = [n] \setminus S$.

Let $\omega_S \in \Omega_S$. We will write Ω_{ω_S} and $\mu^{(\omega_S)}$ for the ω_S-pinning of Ω and μ_S, where

$$\Omega_{\omega_S} = \{\bar{\omega} \in \Omega[S^c] : \omega_S \oplus \bar{\omega} \in \Omega\} \quad \text{and} \quad \mu^{(\omega_S)}(\bar{\omega}) = \frac{\mu(\omega_S \oplus \bar{\omega})}{\sum_{\tilde{\omega}\in\Omega[S^c]} \mu(\omega_S \oplus \tilde{\omega})}, \tag{6}$$

for all $\omega_S \in \Omega[S]$. The following relation between the S-marginal μ_S and ω_S-pinning is immediate from the definitions. Let, $S, T \subset [n]$ such that $S \cap T = \varnothing$,

$$\mu(\omega_S \oplus \omega_T) = \mu_S(\omega_S) \cdot \mu_T^{(\omega_S)}(\omega_T) \tag{7}$$

We note that the statement of Equation (7) is a simple consequence of the law of conditional probability (Bayes' rule).

The Kullback-Leibler divergence $\mathcal{D}(\mu \parallel \nu)$ between distributions $\mu, \nu \in \triangle_\Omega$ is defined as,

$$\mathcal{D}(\mu \parallel \nu) = \mathop{\mathbb{E}}_{x\sim\nu}\left[\frac{\mu(x)}{\nu(x)} \log \frac{\mu(x)}{\nu(x)}\right] \tag{KL divergence}$$

where we have adopted the convention $0/0 = 1$ and assumed $\mathrm{supp}(\mu) \subset \mathrm{supp}(\nu)$.

It is a well-known fact that $\mathcal{D}(\bullet \parallel \bullet)$ is monotone-decreasing under the application of a row-stochastic matrix,

▶ **Lemma 15** (Data Processing Inequality). *Let $\mathsf{M} \in \mathbb{R}^{\Omega\times\Omega'}$ be any row-stochastic operator. Then, for any pair of distributions $\mu, \nu \in \Delta_\Omega$ we have,*

$$\mathcal{D}(\mu\mathsf{M} \parallel \nu\mathsf{M}) \leq \mathcal{D}(\mu \parallel \nu).$$

We also recall the following consequence of the chain-rule for the KL divergence,

▶ **Lemma 16.** *Let $n \geq 0$ finite sets $\Omega_1, \ldots, \Omega_n$ be given. Suppose the set $\Omega \subset \Omega_1 \times \cdots \times \Omega_n$ and the distribution $\mu \in \triangle_\Omega$ are arbitrary. Then, writing $S^c = [n] \setminus S$, we have*

$$\mathcal{D}(\mu \parallel \nu) = \mathcal{D}(\mu_S \parallel \nu_S) + \mathop{\mathbb{E}}_{\omega_S\sim\mu_S} \mathcal{D}(\mu_{S^c}^{(\omega_S)} \parallel \nu_{S^c}^{(\omega_S)}).$$

2.3 Random Walks and Mixing Times

The mixing time $T(\varepsilon, \mathsf{P})$ of the random walk operator $\mathsf{P} \in \mathbb{R}^{V\times V}$ is defined to be the least time step where the distribution of the random walk is ε-close to the stationary distribution π of P in the ℓ_1 distance, i.e. $\pi^{(t)}$ for the distribution of the random walk after t-steps, we have

$$T(\varepsilon, \mathsf{P}) = \min\Big\{t \in \mathbb{N}_{\geq 0} : \|\pi^{(t)} - \pi\|_{\ell_1} \leq \varepsilon\Big\}.$$

We will write $\mathsf{A}(x, \bullet)$ for the x-th row of the matrix A. Recalling that the distribution of the random walk P starting from x after t steps is given by $\pi^{(t)} = \mathsf{P}^t(x, \bullet)$, we formally define

$$T(\varepsilon, \mathsf{P}) = \min\big\{t \in \mathbb{N}_{\geq 0} : \|\mathsf{P}^t(x, \bullet) - \pi^\top\|_{\ell_1} \leq \varepsilon \text{ for all } x \in V\big\}. \tag{mixing time}$$

We define the spectral gap $\mathsf{gap}(\mathsf{P})$ of the random walk $\mathsf{P} \in \mathbb{R}^{\Omega\times\Omega}$ with stationary distribution π as,

$$\mathsf{gap}(\mathsf{P}) = 1 - \sigma_2(\mathsf{P}) \qquad \text{(spectral gap)}$$

The following mixing time bound is well-known, see for example [48, Proposition 1.12].

▶ **Theorem 17** (Spectral Mixing Time Bound). *Let $\mathsf{P} \in \mathbb{R}^{V\times V}$ be a random walk matrix P with stationary distribution π. One has,*

$$T(\varepsilon,\mathsf{P}) \le \left\lceil \frac{\log\left(\left(\varepsilon\cdot\sqrt{\min_{x\in V}\pi(x)}\right)^{-1}\right)}{\log(1/\sigma_2(\mathsf{P}))} \right\rceil \le \left\lceil \frac{\log\left(\left(\varepsilon\cdot\sqrt{\min_{x\in V}\pi(x)}\right)^{-1}\right)}{\mathsf{gap}(\mathsf{P})} \right\rceil$$

where $\mathsf{gap}(\mathsf{P})$ is the spectral gap of the operator P.

The entropy contraction factor $\mathsf{ec}(\mathsf{P})$ of the random walk operator P with stationary measure π is defined as,

$$\mathsf{ec}(\mathsf{P}) = 1-\kappa(\mathsf{P}) \ \text{ where } \ \kappa(\mathsf{P}) = \sup\left\{ \frac{\mathcal{D}(\mu\mathsf{P} \parallel \pi)}{\mathcal{D}(\mu \parallel \pi)} : \mu \in \triangle_\Omega \right\} \qquad \text{(entropy contraction factor)}$$

Then, the following estimate can be used to bound the mixing time, see for example [13, Lemma 2.4],

▶ **Theorem 18.** *Let $\mathsf{P} \in \mathbb{R}^{V\times V}$ be a random walk operator P with stationary distribution π. One has,*

$$T(\varepsilon,\mathsf{P}) \le \frac{C}{\mathsf{ec}(\mathsf{P})} \cdot \log\log \frac{1}{\varepsilon\cdot\min_{x\in V}\pi(x)},$$

where $\mathsf{ec}(\mathsf{P})$ is the entropy contraction factor of P and C is a universal constant independent of (P,π).

2.4 (Partite) Simplicial Complexes

A simplicial complex is a downward closed collection of subsets of a finite set U. Formally, $X \subset 2^U$ and whenever $\beta \in X$ for all $\alpha \subset \beta$ we have $\alpha \in X$. The rank of a face α is $|\alpha|$. Given some j, we will adopt the notation $X^{(j)}$ to refer to the collection faces of X of rank j and the notation $X^{(\le j)}$ to refer to the collection of faces of X of rank at most j, i.e.

$$X^{(j)} := X \cap \binom{U}{j} \ \text{ and } \ X^{(\le j)} := \bigcup_{i=0}^{j} X^{(i)}.$$

We say X is a simplicial complex of rank n if the largest rank of any face $\alpha \in X$ is n. We note that by definition $X^{(0)} = \{\varnothing\}$.

We say that a simplicial complex X of rank n is pure, if any face $\alpha \in X^{(j)}$ for any $j < n$ is contained in another face $\beta \in X^{(n)}$. Equivalently, in a pure simplicial complex the only inclusion maximal faces are those of maximal rank. In this article, we will only deal with pure simplicial complexes.

A rank-n pure simplicial complex X is called n-partite if we can partition $X^{(1)}$ into disjoint sets $X[1],\dots,X[n]$ such that

$$\text{for all } \beta \in X^{(n)} \text{ and for all } i = 1,\dots,n \ \text{ we have } \ |\beta \cap X[i]| = 1. \qquad (n\text{-partiteness})$$

We will call the sets $X[1], \ldots, X[n]$ the sides of the complex X. Equivalently, every element of a rank-n face $\beta \in X[n]$ comes from a distinct side $X[i]$. We observe that a bipartite graph is a 2-partite simplicial complex.

To keep our nomenclature simple, we will simply refer to a pure n-partite simplicial complex of rank n as an n-partite simplicial complex, i.e. we will not consider n-partite complexes which are not pure.

For a face $\alpha \in X$ we introduce the notation, $\mathrm{type}(\alpha) = \{i \in [n] : \alpha \cap X[i] \neq \varnothing\}$ for the type of the face α, i.e. the collection of sides of X that α intersects.

For any $i \in [n]$ and $\beta \in X^{(n)}$ we will write $\beta_i \in X^{(1)}$ for the unique element of β satisfying $\{\beta_i\} = \beta \cap X[i]$. We will refer to β_i as the i-th coordinate of β. We will also write $\beta_T = \{\beta_t : t \in T\}$ for all $T \subset [n]$. We extend this notation to arbitrary faces $\alpha \in X$ and $T \subset \mathrm{type}(\alpha)$. In keeping with the view that a face $\alpha \in X$ with $\mathrm{type}(\alpha) = \{t_1, \ldots, t_k\}$ can be represented as a tuple $(a_{t_1}, \cdots, a_{t_k})$, we will favour the notation $\alpha \oplus \alpha'$ to denote the union of two faces $\alpha, \alpha' \in X$ with $\mathrm{type}(\alpha) \cap \mathrm{type}(\alpha') = \varnothing$ over the usual notation $\alpha \cup \alpha'$.

We observe that for facets $\beta \in X^{(n)}$, i.e. faces of maximal rank, we have $\mathrm{type}(\beta) = [n]$. Given, $\alpha \in X$ we recall that the link X_α is defined as,

$$X_\alpha = \{(\beta \setminus \alpha) \in X : \beta \in X, \beta \supset \alpha\}.$$

The following observation is immediate,

▶ **Lemma 19.** *Let X be an n-partite simplicial complex with sides $X[1], \ldots, X[n]$ and $\alpha \in X^{(j)}$ for some $j \in [0, n]$. Then, the simplicial complex X_α is an $(n-j)$-partite simplicial complex with sides $X_\alpha[j] := X[j] \cap X_\alpha^{(1)}$ for $j \in [n] \setminus \mathrm{type}(\alpha)$.*

For $T \subset [n]$, we will also introduce the notation $X[T]$ to refer to all faces of X of type T, i.e. $X[T] = \{\alpha \in X : \mathrm{type}(\alpha) = T\}$.

2.5 Weighted Simplicial Complexes

A weighted simplicial complex (X, π) of rank n is a pure simplicial complex of rank n where $\pi := \pi_n$ is a probablity distribution on $X^{(n)}$ with full support, i.e. $\pi \in \triangle_{X^{(n)}}$ and $\mathrm{supp}(\pi) = X^{(n)}$. For $j \in [0, n-1]$, we inductively define the probability distributions $\pi_j : X^{(j)} \to \mathbb{R}$ as

$$\pi_j(\alpha) = \frac{1}{j+1} \sum_{\substack{\beta \in X[j+1], \\ \beta \subset \alpha}} \pi_{j+1}(\beta). \tag{8}$$

Following repeated applications of Equation (8), we obtain the following well-known result (see, e.g. [4, Proposition 2.3.1])

▶ **Proposition 20.** *Let (X, π) be a simplicial complex of rank n. For all $0 \leq j \leq k \leq n$ and $\alpha \in X^{(j)}$, one has*

$$\pi_j(\alpha) = \frac{1}{\binom{k}{j}} \sum_{\substack{\beta \supset \alpha, \\ \beta \in X^{(k)}}} \pi_k(\beta).$$

Similarly, given a face $\alpha \in X^{(j)}$, we define the distribution $\pi^{(\alpha)}$ on $X_\alpha^{(n-j)}$ by conditioning π on the containment of α. Consequently, we have

▶ **Proposition 21.** *Let (X, π) be a simplicial complex of rank n. Let $\alpha \in X^{(j)}$ and $\tau \in X_\alpha^{(\ell)}$ be faces for some $0 \leq \ell \leq j \leq n$. Then,*

$$\pi_l^{(\alpha)}(\tau) = \frac{\pi_{j+\ell}(\alpha \cup \tau)}{\binom{j+\ell}{\ell} \cdot \pi_j(\alpha)}.$$

When (X,π) is an n-partite weighted simplicial complex, for any $T\subset[n]$, we will define π_T as the marginal distribution of π on the coordinates in T according to Equation (5), i.e.

$$\pi_T(\omega_T)=\sum_{\bar\omega\in X[T^c]}\pi(\omega_T\oplus\bar\omega).$$

Similarly, given any $\omega_T\in X[T]$, we define the weighting $\pi^{(\omega_T)}$ on $(X_{\omega_T},\pi^{(\omega_T)})$ according to Equation (6). Thus,

$$\pi^{(\omega_T)}(\bar\omega)=\Pr_{\omega'\sim\pi}[\omega'_{T^c}=\bar\omega\mid\omega'_T=\bar\omega]=\frac{\pi(\omega_T\oplus\bar\omega)}{\pi_T(\omega_T)}\quad\text{for all}\quad\bar\omega\in X[T^c].$$

2.6 Update Operators and the Sequential Sweep

Let (X,π) be a pure, n-partite simplicial complex of rank n. We define the update operators $\mathsf{Q}_1,\dots,\mathsf{Q}_n\in\mathbb{R}^{X^{(n)}\times X^{(n)}}$ as follows,

$$\mathsf{Q}_i(\omega,\omega')=\Pr_{\bar\omega\sim\pi}\left[\bar\omega=\omega'\mid\bar\omega\supset\omega_{[n]\setminus i}\right]=\pi_{\{i\}}^{(\omega_{[n]\setminus i})}(\omega'_i).\tag{9}$$

Thus, Q_i can be thought as describing the random walk which makes a transition from $\omega\in X^{(n)}$ to a random state $\omega'\in X^{(n)}$ by only re-sampling the i-th coordinate from the distribution π without changing the remaining coordinates, $\omega_{[n]\setminus i}$.

Given an ordering $\vec{s}=(s(1),\dots,s(n))$ of $[n]$, we define the sequential sweep $\mathsf{P}_{\mathsf{seq}}^{(\vec{s})}$ as the random walk operator $\mathsf{P}_{\mathsf{seq}}^{(\vec{s})}:=\mathsf{Q}_{s(1)}\cdots\mathsf{Q}_{s(n)}$, which updates the coordinates $s(1),\dots,s(n)$ sequentially one after the other. When $\vec{s}=(1,\dots,n)$, i.e. $\vec{s}$ is the canonical order of $[n]$, we will simply write $\mathsf{P}_{\mathsf{seq}}:=\mathsf{P}_{\mathsf{seq}}^{(\vec{s})}$. We note that the weighting π of the simplicial complex is stationary for $\mathsf{P}_{\mathsf{seq}}$, since $\pi\mathsf{Q}_i=\pi$ for all $i=1,\dots,n$. Thus, $\pi\mathsf{P}_{\mathsf{seq}}=\pi$.

Notice that we have, $\mathsf{P}_{\mathsf{seq}}^*=(\mathsf{Q}_1\cdots\mathsf{Q}_n)^*=\mathsf{Q}_n^*\cdots\mathsf{Q}_1^*=\mathsf{Q}_n\cdots\mathsf{Q}_1$. In particular the random-walk $\mathsf{P}_{\mathsf{seq}}$ is not necessarily reversible!

It has already been observed in [6, 53], that the update operators Q_i are orthogonal projections.

▶ **Proposition 22.** *For all $i=1,\dots,n$ the operator Q_i is an orthogonal projection to the subspace,*

$$\mathcal{U}_i=\operatorname{span}\{\boldsymbol{u}_\alpha:\alpha\in X[1,\cdots,i-1,i+1,\cdots,n]\},$$

where $\boldsymbol{u}_\alpha(\omega)=\mathbf{1}[\omega\supset\alpha]$.

2.7 Colored Random Walks on Partite Simplicial Complexes

It will also be important to recall the colored random walks, introduced in [21, 34]. Let $\alpha\in X$ be any given face of an n-partite weighted simplicial complex. Let $I,J\subset[n]\setminus\operatorname{type}(\alpha)$ be two disjoint sets, i.e. $I\cap J=\varnothing$. We define the (I,J) colored random walk $\mathsf{C}_\alpha^{I\to J}\in\mathbb{R}^{X_\alpha[I]\times X_\alpha[J]}$ from $X_\alpha[I]$ to $X_\alpha[J]$ by setting

$$\mathsf{C}_\alpha^{I\to J}(\tau_I,\tau_J)=\pi_J^{(\alpha\oplus\tau_I)}(\tau_J)=\Pr_{\omega\sim\pi}[\omega_J=\tau_J\mid\omega_I=\tau_I,\omega_{\operatorname{type}(\alpha)}=\alpha]\tag{10}$$

for all $\tau_I\in X_\alpha[I]$ and $\tau_J\in X_\alpha[J]$.

Let $I, J \subset [n]$ be two subsets satisfying $I \cap J = \varnothing$. Suppose for some $\alpha \in X[(I \cup J)^c]$ the marginal distributions $\pi_I^{(\alpha)}$ and $\pi_J^{(\alpha)}$ are independent, i.e. we have

$$\pi_{I\cup J}^{(\alpha)}(\tau_I \oplus \tau_J) = \pi_I^{(\alpha)}(\tau_I) \cdot \pi_J^{(\alpha)}(\tau_J) \text{ for all } \tau_I \in X_\alpha[I], \tau_J \in X_\alpha[J].$$

It is easy to see, that in this case $\mathsf{C}_\alpha^{I\to J} = \mathbf{1}\pi_J^{(\alpha)}$. In particular, we have[9]

$$\sigma_2(\mathsf{C}_\alpha^{I\to J}) = \sigma_2(\mathbf{1}\pi_J^{(\alpha)}) = 0 \text{ and } \mathcal{D}(\mu\mathsf{C}_\alpha^{I\to J} \parallel \nu\mathsf{C}_\alpha^{I\to J}) = \mathcal{D}(\pi_J^{(\alpha)} \parallel \pi_J^{(\alpha)}) = 0,$$

for all $\mu, \nu \in \triangle_{X_\alpha[I]}$. We will consider relaxations of both statements to consider the proximity of $\pi_I^{(\alpha)}$ and $\pi_J^{(\alpha)}$ to being product distributions. Thus, the (I, J) variance contraction parameter $\varepsilon^{I\to J}$ is defined as,

$$\varepsilon^{I\to J} := \varepsilon_{(X,\pi)}^{I\to J} = \max_{\alpha\in X[(I\cup J)^c]} \sigma_2\big(\mathsf{C}_\alpha^{I\to J}\big) \tag{11}$$

We also define, the related (I, J) entropy contraction parameter

$$\eta^{I\to J} = 1-\kappa^{I\to J} \;;\; \kappa^{I\to J} = \sup\left\{ \frac{\mathcal{D}(\mu\mathsf{C}_\alpha^{I\to J} \parallel \pi_J^{(\alpha)})}{\mathcal{D}(\mu \parallel \pi_I^{(\alpha)})} : \alpha \in X[(I \cup J)^c] \text{ and } \mu \in \triangle_{X_\alpha[I]} \right\} \tag{12}$$

$\vec{\varepsilon}$-product distributions and $\varepsilon^{I\to J}$

When dealing with the colored random walks $\mathsf{C}_\alpha^{I\to J}$ an alternative notion of high-dimensional expansion introduced in [34] proves more convenient: Let (X, π) be an n-partite simplicial complex with parts $X[1], \cdots, X[n]$. The distribution π is called ε-product if,

$$\sigma_2(\mathsf{C}_\varnothing^{i\to j}) \le \varepsilon \text{ for all } i, j \in [n] \text{ where } i \neq j.$$

We extend this definition in the following manner: Let $\vec{\varepsilon} = (\varepsilon_0, \ldots, \varepsilon_{n-2})$ and (X, π) be an arbitrary n-partite simplicial complex. We say the distribution π is $\vec{\varepsilon}$-product if for all $\alpha \in X^{(\le n-2)}$ we have,

$$\sigma_2(\mathsf{C}_\alpha^{i\to j}) \le \varepsilon_{|\alpha|} \text{ for all } i, j \in [n] \setminus \alpha \text{ where } i \neq j.$$

Equivalently, π is $\vec{\varepsilon}$-product if and only if $\pi^{(\alpha)}$ is $\varepsilon_{|\alpha|}$-product for all $\alpha \in X^{(\le n-2)}$. We recall the following result of [21, Claim 7.10],

▶ **Proposition 23.** *Let (X, π) be an n-partite simplicial complex in which every link G_α is connected – equivalently, (X, π) is a one-sided $\vec{\gamma}$-local spectral expander where $\gamma_i < 1$ for all i. Then, the bipartite graph $G_{\{i\},\{j\}}$ described by $\mathsf{C}_\alpha^{i\to j}$ is connected. In particular, there exists a vector $\vec{\varepsilon} = (\varepsilon_i)_{i=0}^{n-2}$ such that $\varepsilon_i < 1$ for all i, for which the distribution π is $\vec{\varepsilon}$-product.*

They also prove the following down-trickling result, [21, Corollary 7.6]

▶ **Theorem 24.** *Let (X, π) be an n-partite simplicial complex all of whose links are connected. Suppose $\gamma_{n-2}(X, \pi) \le \frac{\varepsilon}{(n-2)\varepsilon+1}$. Then, for every $\alpha \in X^{(k)}$ and every $i, j \in [n] \setminus \mathrm{type}(\alpha)$, we have $\sigma_2(\mathsf{C}_\alpha^{i\to j}) \le \frac{\varepsilon}{k\varepsilon+1}$.*

[9] the first equality is the consequence of $\mathsf{C}_\alpha^{I\to J}$ being a rank one matrix

3 A Spectral Gap Bound for $\mathsf{P}_{\mathsf{seq}}$

Our main theorem in this section will be,

▶ **Theorem 25.** *Let (X,π) be an n-partite simplicial complex, $\vec{s}=(s(1),\dots,s(n))$ be any ordering of $[n]$, and the parameters $\varepsilon^{I\to J}$ be defined as in Equation (11) for all $I,J\subset[n]$ with $I\cap J=\varnothing$. Then,*

$$\sigma(\mathsf{P}_{\mathsf{seq}}^{(\vec{s})})^2 \le 1-\prod_{j=2}^{n}\Big(1-(\varepsilon^{s([j-1])\to s(j)})^2\Big),$$

where we have written $S(A)=\{s(a):a\in S\}$. Equivalently,

$$\mathsf{gap}(\mathsf{P}_{\mathsf{seq}}^{(\vec{s})}) \ge 1-\sqrt{1-\prod_{j=2}^{n}\big(1-(\varepsilon^{s([j-1])\to s(j)})^2\big)}.$$

In particular, if π is $\vec{\varepsilon}$-product, we have

$$\sigma(\mathsf{P}_{\mathsf{seq}}^{(\vec{s})})^2 \le 1-\prod_{j=2}^{n}\prod_{p=0}^{j-2}(1-\varepsilon_p^2).$$

Equivalently,

$$\mathsf{gap}(\mathsf{P}_{\mathsf{seq}}^{(\vec{s})}) \ge 1-\sqrt{1-\prod_{j=2}^{n}\prod_{p=0}^{j-2}(1-\varepsilon_p)^2}$$

In conjunction with Theorem 24, Theorem 25 immediately implies the following

▶ **Corollary 26.** *Let (X,π) be a link-connected n-partite simplicial complex that is a $\vec{\gamma}$-local spectral expander with $\gamma_i\le\varepsilon/(n-1-i)$ (i.e. π $\vec{\varepsilon}$-product, with $\varepsilon_i\le\varepsilon$), then*

$$\sigma_2(\mathsf{P}_{\mathsf{seq}})\le\frac{n\cdot\varepsilon}{\sqrt{2}} \quad \textit{and} \quad \mathsf{gap}(\mathsf{P}_{\mathsf{seq}})\ge 1-\frac{n\cdot\varepsilon}{\sqrt{2}}.$$

If we have $\gamma_{n-2}\le\frac{\varepsilon}{2n}$ and all links of (X,π) are connected, then

$$\sigma_2(\mathsf{P}_{\mathsf{seq}})\le\frac{\varepsilon}{\sqrt{2}} \quad \textit{and} \quad \mathsf{gap}(\mathsf{P}_{\mathsf{seq}})\ge 1-\frac{\varepsilon}{\sqrt{2}}.$$

For the second statement we note that Theorem 24 implies that we are completely ε/n product assuming $\gamma_{n-2}\le\varepsilon/2n$. In particular, the ε in the statement of Theorem 24 and the ε above differ by a factor of n. The above corollary shows that for constant n, having constant local spectral expansion guarentees that $\mathsf{gap}(\mathsf{P}_{\mathsf{seq}})$ is close to 1 – thus it allows us to conclude that $\mathsf{P}_{\mathsf{seq}}$ mixes rapidly in the complexes of [47, 41].

We will prove Theorem 25 using Theorem 14. To be able apply Theorem 14, we will need some preparatory work:

1. For all $T\subset[n]$, we will characterize the intersections $\bigcap_{t\in T}\mathcal{U}_t$, where $\mathcal{U}_t$ are as defined in Proposition 22, i.e. $\mathcal{U}_t=\mathrm{im}(\mathsf{Q}_t)$. This will be done in Proposition 27.
2. We will get a bound on $\sin^2(\mathcal{U}_p,\bigcap_{t\in T}\mathcal{U}_t)$ for all $p\in[n]$ and $T\subset[n]\setminus p$ in terms of the parameters $\varepsilon^{I\to J}$, defined in Equation (11). This will be done in Theorem 29.
3. We will get a bound on the parameters $\varepsilon^{I\to J}$ under the assumption that π is $\vec{\varepsilon}$-product. This will be done in Theorem 30.

We state these results now.

▶ **Proposition 27.** *For (X,π) a weighted and link-connected n-partite simplicial complex. Let $\mathcal{U}_1,\dots,\mathcal{U}_n$ be as defined in Proposition 22. For all $T\subset[n]$, we define*

$$\mathcal{U}_T := \operatorname{span}\{\boldsymbol{u}_\alpha : \alpha\in X[T^c]\}.$$

Then, $\mathcal{U}_T=\bigcap_{t\in T}\mathcal{U}_t$.

▶ Remark 28. We have,

$$\mathcal{U}_I=\operatorname{span}\Big\{\boldsymbol{f}\in\mathbb{R}^{X^{(n)}} : \boldsymbol{f}(\omega)=\boldsymbol{f}(\bar\omega) \text{ for all } \omega,\bar\omega\in X^{(n)} \text{ such that } \omega_{[n]\setminus I}=\bar\omega_{[n]\setminus I}\Big\}.$$

In particular, $\mathcal{U}_I$ is the space of vectors $\boldsymbol{f}\in\mathbb{R}^{X^{(n)}}$ whose entries are completely determined by the coordinates in $[n]\setminus I$.

We note that a special case of this result was already proven in [53, Proposition 3.6]. We will present a proof for Proposition 27, in Section 3.2.

Then, we show the geometric interpretation for the quantities $\varepsilon^{I\to J}$ as follows,

▶ **Theorem 29.** *Let (X,π) be an n-partite simplicial complex. Let $I,J\subset[n]$ be any pair of disjoint sets, i.e. $I\cap J=\varnothing$.*

Then, $\cos(\mathcal{U}_I,\mathcal{U}_J)\le\varepsilon^{I\to J}$, where the subspaces $\mathcal{U}_I,\mathcal{U}_J$ are as defined in Proposition 27 and the parameters $\varepsilon^{I\to J}$ are as defined in Equation (11).

We will prove Theorem 29 in Section 3.3.

Finally, we state our bound on the parameters $\varepsilon^{I\to J}$,

▶ **Theorem 30.** *Let (X,π) be an n-partite simplicial complex where π is $\vec{\varepsilon}$-product. We have,*

$$\sigma_2(\mathsf{C}_\alpha^{I\to J})^2\le 1-\prod_{p=0}^{|I|-1}\prod_{q=0}^{|J|-1}(1-\varepsilon^2_{|\alpha|+p+q}) \quad \textit{and thus} \quad (\varepsilon^{I\to J})^2\le 1-\prod_{p=0}^{|I|-1}\prod_{j=0}^{|J|-1}(1-\varepsilon^2_{n-|I|-|J|+p+q}) \tag{13}$$

▶ Remark 31. Notice that the RHS of Equation (13) is always strictly smaller than 1 provided that π is $\vec{\varepsilon}$-product with $\varepsilon_i<1$ for all $i=0,\dots,n-2$. By Proposition 23, this is always the case assuming all the links in (X,π) are connected.

We will present the proof for Theorem 30 in Section 3.1.

The proof of Theorem 25 is now immediate from Theorem 14.

3.1 Proof of Theorem 30

Proof of Theorem 30. If $|I|=|J|=1$, then the RHS of Equation (13) equals $\varepsilon_{|\alpha|}$ and thus the statement follows since π is $\vec{\varepsilon}$-product. Suppose without loss of generality, that $\alpha=\varnothing$ and $|I|\ge 2$ where we have some fixed element $i\in I$. Write,

$$\sigma := \max_{x\in X[i]}\sigma_2\Big(\mathsf{C}_x^{(I\setminus i)\to J}\Big). \tag{14}$$

Let $\boldsymbol{F}\in X[J]\to\mathbb{R}$ satisfy $\|\boldsymbol{F}\|_{\pi_J}=1$, $\langle\boldsymbol{F},\mathbf{1}\rangle_{\pi_J}=0$, and $\|\mathsf{C}_\varnothing^{I\to J}\boldsymbol{F}\|_{\pi_I}=\sigma_2(\mathsf{C}_\varnothing^{I\to J})$.

Then,

$$\begin{aligned}
\sigma_2\left(\mathsf{C}_{\varnothing}^{I\to J}\right)^2 = \left\|\mathsf{C}_{\varnothing}^{I\to J}\boldsymbol{F}\right\|_{\pi_I}^2 &= \mathop{\mathbb{E}}_{\alpha_I\sim\pi_I}\left(\mathop{\mathbb{E}}_{\alpha_J\sim\pi_J^{(\alpha_I)}}\boldsymbol{F}(\alpha_J)\right)^2,\\
&= \mathop{\mathbb{E}}_{x\sim\pi_{\{i\}}}\mathop{\mathbb{E}}_{\alpha'\sim\pi_{I\setminus i}^{(x)}}\left(\mathop{\mathbb{E}}_{\alpha_J\sim\pi_J^{(x\oplus\alpha')}}\boldsymbol{F}(\alpha_J)\right)^2,\\
&= \mathop{\mathbb{E}}_{x\sim\pi_{\{i\}}}\left\|\mathsf{C}_x^{(I\setminus i)\to J}\boldsymbol{F}_x\right\|_{\pi_{I\setminus i}^{(x)}}^2,\\
&\le \mathop{\mathbb{E}}_{x\sim\pi_{\{i\}}}\left[\left\|\boldsymbol{F}_x^{\mathbf{1}}\right\|_{\pi_J^{(x)}}^2 + \sigma_2(\mathsf{C}_x^{(I\setminus i)\to J})^2\cdot\left\|\boldsymbol{F}_x^{\perp\mathbf{1}}\right\|_{\pi_J^{(x)}}^2\right],\\
&= \sigma^2 + (1-\sigma^2)\cdot\mathop{\mathbb{E}}_{x\sim\pi_{\{i\}}}\|\boldsymbol{F}_x^{\mathbf{1}}\|_{\pi_J^{(x)}}^2
\end{aligned}$$

where $\boldsymbol{F}_x^{\mathbf{1}} = \mathbf{1}\cdot\mathbb{E}_\alpha\,\boldsymbol{F}_x(\alpha_{J\setminus i})$. We note now for each $i\in X[i]$,

$$\left\|\boldsymbol{F}_x^{\mathbf{1}}\right\|_{\pi_J}^2 = \left(\mathop{\mathbb{E}}_{\alpha_J\sim\pi_J^{(x)}}\boldsymbol{F}(\alpha_J)\right)^2 = [\mathsf{C}_{\varnothing}^{i\to J}\boldsymbol{F}](x)^2.$$

Thus, using that $\langle\boldsymbol{F},\mathbf{1}\rangle_{\pi_J}=0$,

$$\mathop{\mathbb{E}}_{x\sim\pi_{\{1\}}}\left\|\boldsymbol{F}_x^{\mathbf{1}}\right\|_{\pi_J}^2 = \|\mathsf{C}_{\varnothing}^{i\to J}\boldsymbol{F}\|_{\pi_{\{i\}}}^2 \le \sigma_2\left(\mathsf{C}_{\varnothing}^{i\to J}\right)^2.$$

Thus,

$$\begin{aligned}
\sigma_2\left(\mathsf{C}_{\varnothing}^{I\to J}\right)^2 &\le 1-(1-\sigma^2)(1-\sigma_2(\mathsf{C}_{\varnothing}^{i\to J})^2),\\
&= 1-\left(1-\sigma_2\left(\mathsf{C}_{x^\star}^{I\setminus i\to J}\right)^2\right)\left(1-\sigma_2\left(\mathsf{C}_{\varnothing}^{i\to J}\right)^2\right).
\end{aligned}\tag{15}$$

where $x^\star = \arg\max_{x\in X[i]}\sigma_2(\mathsf{C}_x^{I\setminus i\to J})$.

Now, we first assume $J=\{j\}$ and $I=\{i_1,\dots,i_\ell\}$, applying Equation (15) recursively, we obtain

$$\sigma_2(\mathsf{C}_{\varnothing}^{I\to\{j\}}) \le 1-(1-\sigma_2(\mathsf{C}_{\varnothing}^{i_1\to j})^2)\prod_{p=k}^{\ell}\left(1-\sigma_2\left(\mathsf{C}_{x_{i_1}\oplus\cdots\oplus x_{i_{p-1}}}^{i_p\to\{j\}}\right)^2\right),$$

where

$$x_{i_1}=x^\star,\quad \omega(p)=x_{i_1}\oplus\cdots x_{i_{p-1}},\quad\text{and}\quad x_p=\arg\max_{x\in X_{\omega(p)}[i_p]}\sigma_2\left(\mathsf{C}_{x_{i_1}\oplus\cdots\oplus x_{i_{p-1}}\oplus x}^{I\setminus\{i_1,\dots,i_p\}\to\{j\}}\right),$$

In particular using the $\vec{\varepsilon}$-product assumption,

$$\sigma_2(\mathsf{C}_{\varnothing}^{I\to j})^2 \le 1-\prod_{p=0}^{|I|-1}(1-\varepsilon_p^2).$$

Now, by using $\mathsf{C}_{\varnothing}^{J\to i}=\left(\mathsf{C}_{\varnothing}^{i\to J}\right)^*$ and that $\sigma_2(\mathsf{B})=\sigma_2(\mathsf{B}^*)$ we have by the above

$$\sigma_2(\mathsf{C}_{\varnothing}^{i\to J})^2 \le 1-\prod_{p=0}^{|J|-1}(1-\varepsilon_p^2).\tag{16}$$

Similarly, inserting Equation (16) inside Equation (15), we obtain

$$\sigma_2\big(\mathsf{C}_{\varnothing}^{I\to J}\big)^2 \le 1 - \prod_{p=0}^{|J|-1}(1-\varepsilon_p^2)\cdot\left(1-\sigma_2\Big(\mathsf{C}_{x^\star}^{I\setminus i\to J}\Big)^2\right). \tag{17}$$

Suppose $I=\{i_1,\dots,i_\ell\}$ write $x_{i_1}=x^\star$ and

$$\omega(q)=x_{i_1}\oplus\cdots x_{i_{q-1}},\quad\text{and}\quad x_q=\arg\max_{x\in X_{\omega(q)}[i_q]}\sigma_2\Big(\mathsf{C}_{\omega(q)\oplus x}^{I\setminus\{i_1,\dots,i_p\}\to J}\Big),$$

Applying Equation (17) repeatedly, we obtain

$$\sigma_2(\mathsf{C}_{\varnothing}^{I\to J})^2 \le 1-\left(\prod_{p=0}^{|J|-1}(1-\varepsilon_p^2)\right)\cdot\prod_{q=1}^{|I|-1}\left(1-\sigma_2\Big(\mathsf{C}_{x_{i_1}\oplus\cdots\oplus x_{i_q}}^{i_{q+1}\to J}\Big)^2\right). \tag{18}$$

By Equation (16), we have

$$\sigma_2\Big(\mathsf{C}_{x_{i_1}\oplus\cdots\oplus x_{i_q}}^{i_{q+1}\to J}\Big)^2 \le 1-\prod_{p=0}^{|J|-1}(1-\varepsilon_{q+p}^2)$$

And inserting this in Equation (18), we obtain

$$\begin{aligned}\sigma_2\big(\mathsf{C}_{\varnothing}^{I\to J}\big)^2 &\le 1-\left(\prod_{p=0}^{|J|-1}(1-\varepsilon_p^2)\right)\cdot\left(\prod_{p=0}^{|J|-1}(1-\varepsilon_{p+1}^2)\right)\cdots\left(\prod_{p=0}^{|J|-1}(1-\varepsilon_{p+|I|-1}^2)\right),\\ &= 1-\prod_{q=0}^{|I|-1}\prod_{p=0}^{|I|-1}(1-\varepsilon_{p+q}^2).\end{aligned}$$

◀

3.2 Proof of Proposition 27

Proof of Proposition 27. We proceed by induction on the number of elements in T. For $|T|=1$, the statement is clear. Thus, we assume that

$$\mathcal{U}_T=\bigcap_{t\in T}\mathcal{U}_t\quad\text{for all}\quad T\quad\text{with}\quad |T|=\ell.$$

Let $t\in[n]\setminus T$, and consider $T\cup\{t\}$. First, fix any $\alpha\in X[[n]\setminus(T\cup t)]$. Then, note

$$\boldsymbol{u}_\alpha=\sum_{\substack{\beta\in X[[n]\setminus T],\\ \beta\supset\alpha}}\boldsymbol{u}_\beta=\sum_{\substack{\beta'\in X[1,\dots,t-1,t+1,\dots,n],\\ \beta'\supset\alpha}}\boldsymbol{u}_{\beta'}.$$

Thus, indeed: $\operatorname{span}\{\boldsymbol{u}_\alpha:\alpha\in X[[n]\setminus(T\cup t)]\}=\mathcal{U}_{T\cup t}\subset\mathcal{U}_T\cap\mathcal{U}_t$.

To show the reverse inclusion, we recall that a function $\boldsymbol{f}\in\mathbb{R}^{X^{(n)}}$ is in $\mathcal{U}_A$ if and only if we have

$$\boldsymbol{f}(\omega_A\oplus\omega_{[n]\setminus A})=\boldsymbol{f}(\tilde{\omega}_A\oplus\omega_{[n]\setminus A})$$

for all $\omega_A,\tilde{\omega}_A\in X[A]$ and $\omega_{[n]\setminus A}\in X[[n]\setminus A]$ such that $\omega_A\oplus\omega_{[n]\setminus A},\tilde{\omega}_A\oplus\omega_{[n]\setminus A}\in X^{(n)}$.

i.e. the value of $\boldsymbol{f}$ is determined only by the coordinates in $[n]\setminus A$ (qv. Remark 28).

Suppose $\boldsymbol{f} \in \mathcal{U}_T \cap \mathcal{U}_t$. We want to prove,

$$\boldsymbol{f}(\omega_t \oplus \omega_T \oplus \omega_{[n]\setminus(T\cup t)}) = \boldsymbol{f}(\tilde{\omega}_t \oplus \tilde{\omega}_T \oplus \omega_{[n]\setminus(T\cup t)}), \tag{19}$$

for all $\omega_t, \tilde{\omega}_t \in X[t]$, $\omega_T, \tilde{\omega}_T \in X[T]$ and $\omega_{[n]\setminus(T\cup t)} \in X[(T\cup t)^c]$ such that $\omega_t \oplus \omega_T \oplus \omega_{(T\cup t)^c}, \tilde{\omega}_t \oplus \tilde{\omega}_T \oplus \omega_{(T\cup t)^c} \in X^{(n)}$.

Since $\boldsymbol{f} \in \mathcal{U}_t$, we have,

$$\boldsymbol{f}(\omega_t \oplus \omega_T \oplus \omega_{[n]\setminus(T\cup t)}) = \boldsymbol{f}(\tilde{\omega}_t \oplus \omega_T \oplus \omega_{[n]\setminus(T\cup t)})$$

for all $\omega_t, \tilde{\omega}_t \in X[t]$, $\omega_T \in X[T]$, and $\omega_{[n]\setminus(T\cup t)} \in X[[n]\setminus(T\cup t)]$ such that $\omega_t \oplus \omega_T \oplus \omega_{[n]\setminus(T\cup t)}, \tilde{\omega}_t \oplus \omega_T \oplus \omega_{[n]\setminus(T\cup t)} \in X^{(n)}$. Similarly, since $\boldsymbol{f} \in \mathcal{U}_T$, we have,

$$\boldsymbol{f}(\omega_t \oplus \omega_T \oplus \omega_{[n]\setminus(T\cup t)}) = \boldsymbol{f}(\omega_t \oplus \tilde{\omega}_T \oplus \omega_{[n]\setminus(T\cup t)})$$

for all $\omega_t \in X[t]$, $\omega_T, \tilde{\omega}_T \in X[T]$, and $\omega_{[n]\setminus(T\cup t)} \in X[[n]\setminus(T\cup t)]$ such that $\omega_t \oplus \omega_T \oplus \omega_{[n]\setminus(T\cup t)}, \omega_t \oplus \tilde{\omega}_T \oplus \omega_{[n]\setminus(T\cup t)} \in X^{(n)}$. Fixing $\omega_{[n]\setminus(T\cup t)}$, this means that Equation (19) is satisfied by any pair $(\omega_t \oplus \omega_T, \tilde{\omega}_t \oplus \tilde{\omega}_T)$ which is connected by an edge in the line graph[10] $L(G_{t,T}^{(\omega_{[n]\setminus(T\cup t)})})$, where $G_{t,T}^{(\omega_{[n]\setminus(T\cup t)})}$ is the graph underlying $\mathsf{C}_{\omega_{[n]\setminus(T\cup t)}}^{t\to T}$. By Remark 31, since the links of (X,π) are connected, so is the graph $G_{t,T}^{(\omega_{[n]\setminus(T\cup t)})}$ for any $\omega_{[n]\setminus(T\cup t)} \in X[[n]\setminus(T\cup t)]$. In particular, the graph $L(G_{t,T}^{(\omega_{[n]\setminus(T\cup t)})})$ is connected, meaning any pair of $\omega_t \oplus \omega_T, \tilde{\omega}_t \oplus \tilde{\omega}_T \in X_{\omega_{[n]\setminus(T\cup t)}}[T\cup t]$ are connected by a path and therefore satisfy Equation (19). This allows us to conclude $\boldsymbol{f} \in \mathcal{U}_{T\cup t}$. ◀

3.3 Proof of Theorem 29

Proof. By Proposition 27 we have $\mathcal{U}_I \cap \mathcal{U}_J = \mathcal{U}_{I\cup J}$ and by Equation (4) we have,

$$\cos(\mathcal{U}_I, \mathcal{U}_J) = \max\left\{\langle \boldsymbol{h}, \boldsymbol{g}\rangle_\pi : \|\boldsymbol{h}\|_\pi = \|\boldsymbol{g}\|_\pi = 1 \text{ and } \boldsymbol{g} \in \mathcal{U}_I \cap \mathcal{U}_{I\cup J}^\perp, \boldsymbol{h} \in \mathcal{U}_J \cap \mathcal{U}_{I\cup J}^\perp\right\}. \tag{20}$$

Now, let $\boldsymbol{h}$ and $\boldsymbol{g}$ be the optimizers of the formula in Equation (20). For convenience we introduce the functions $\boldsymbol{G} : X[[n]\setminus I] \to \mathbb{R}$ and $\boldsymbol{H} : X[[n]\setminus J] \to \mathbb{R}$ where,

$$\begin{aligned}\boldsymbol{h} &:= \sum_{\alpha \in X[[n]\setminus J]} \boldsymbol{H}(\alpha)\cdot \boldsymbol{u}_\alpha,\\ \boldsymbol{g} &:= \sum_{\bar{\alpha} \in X[[n]\setminus I]} \boldsymbol{G}(\bar{\alpha})\cdot \boldsymbol{u}_{\bar{\alpha}}.\end{aligned}$$

where the vectors $\boldsymbol{u}_\bullet$ are as defined in Proposition 22, i.e. $\boldsymbol{u}_\tau(\omega) = \mathbf{1}[\omega \supset \tau]$ for all $\tau \in X$. We observe that both $\boldsymbol{H}$ and $\boldsymbol{G}$ have the same norm as $\boldsymbol{h}$ and $\boldsymbol{g}$ respectively.

▷ Claim 32. We have,

$$\|\boldsymbol{H}\|_{\pi_{[n]\setminus J}} = \|\boldsymbol{G}\|_{\pi_{[n]\setminus I}} = 1.$$

Proof. Since the proofs are identical, we only prove the statement for $\boldsymbol{H}$. By the orthogonality of the collection $\boldsymbol{u}_\bullet$ we have,

$$\|\boldsymbol{h}\|_\pi^2 = \sum_{\alpha \in X[[n]\setminus J]} \boldsymbol{H}^2(\alpha)\cdot \|\boldsymbol{u}_\alpha\|_\pi^2.$$

[10] The line graph $L(G)$ of a simple graph G is the graph $L(G) = (E, E')$ where $ef \in E'$ if $|e\cap f| = 1$, i.e. there is an edge $ef \in E'$ for any pair of distinct edges sharing an end point.

Notice, for any $\alpha \in X[[n] \setminus J]$,

$$\|\boldsymbol{u}_\alpha\|_\pi^2 = \sum_{\substack{\beta \in X^{(n)}, \\ \beta \supset \alpha}} \pi(\beta) = \pi_{[n]\setminus J}(\alpha),$$

where we have used the definition of $\pi_{[n]\setminus J}$, i.e. Equation (5) for this equality.

In particular,

$$\|\boldsymbol{h}\|_\pi^2 = \sum_{\alpha \in X[[n]\setminus J]} \boldsymbol{H}(\alpha)^2 \cdot \pi_{[n]\setminus J}(\alpha) = \|\boldsymbol{H}\|_{\pi_{[n]\setminus J}}.$$

Since $\boldsymbol{h}$ is picked as an optimizer to Equation (20), we have $\|\boldsymbol{H}\|_{\pi_{[n]\setminus J}} = 1$. ◁

Then, note

$$\begin{aligned}\langle \boldsymbol{h}, \boldsymbol{g}\rangle_\pi &= \sum_{\alpha \in X[[n]\setminus J]} \sum_{\bar\alpha \in X[[n]\setminus I]} \boldsymbol{H}(\alpha)\boldsymbol{G}(\bar\alpha) \cdot \langle \boldsymbol{u}_\alpha, \boldsymbol{u}_{\bar\alpha}\rangle_\pi, \\ &= \sum_{\alpha \in X[[n]\setminus J]} \sum_{\bar\alpha \in X[[n]\setminus I]} \boldsymbol{H}(\alpha)\boldsymbol{G}(\bar\alpha) \cdot \pi_n(\alpha \wedge \bar\alpha).\end{aligned}$$

where we define $\alpha \wedge \bar\alpha$ as $\perp$[11] if there exists some $\ell \in [n] \setminus (I \cup J)$ such that $\alpha_\ell \neq (\bar\alpha)_\ell$ and otherwise we have,

$$(\alpha \wedge \bar\alpha)_i = \begin{cases} \alpha_i & \text{if } i \in I, \\ \bar\alpha_i & \text{if } i \in J, \\ \alpha_i = (\bar\alpha)_i & \text{if } i \in [n] \setminus (I \cup J) \end{cases}$$

In particular, since the $(\alpha, \bar\alpha)$-term is only non-$\perp$ when α and $\bar\alpha$ agree on the coordinates $[n]\setminus(I\cup J)$, we can localize this sum at the intersections $\tilde\alpha \in X[[n]\setminus(I\cup J)]$. In particular, writing $\boldsymbol{H}_{\tilde\alpha}(\alpha_I) = \boldsymbol{H}(\tilde\alpha \oplus \alpha_I)$ and $\boldsymbol{G}_{\tilde\alpha}(\alpha_J) = \boldsymbol{G}(\tilde\alpha \oplus \alpha_J)$ for $\alpha_I \in X_{\tilde\alpha}[I]$ and $\alpha_J \in X_{\tilde\alpha}[J]$, we have

$$\begin{aligned}\langle \boldsymbol{h}, \boldsymbol{g}\rangle_\pi &= \sum_{\alpha \in X[[n]\setminus J]} \sum_{\bar\alpha \in X[\tilde\alpha \in X[[n]\setminus I]]} \boldsymbol{H}(\alpha)\boldsymbol{G}(\bar\alpha) \cdot \pi_n(\alpha \wedge \bar\alpha), \\ &= \sum_{\tilde\alpha \in X[[n]\setminus(I\cup J)]} \left(\sum_{\alpha_I \in X_{\tilde\alpha}[I]} \sum_{\alpha_J \in X_{\tilde\alpha}[J]} \boldsymbol{H}_{\tilde\alpha}(\alpha_I)\boldsymbol{G}_{\tilde\alpha}(\alpha_J)\pi(\alpha_I \oplus \alpha_J \oplus \tilde\alpha) \right), \\ &=^{\blacklozenge} \sum_{\tilde\alpha \in X[[n]\setminus(I\cup J)]} \pi_{[n]\setminus(I\cup J)}(\tilde\alpha) \cdot \left(\sum_{\alpha_J \in X_{\tilde\alpha}[J]} \sum_{\alpha_I \in X_{\tilde\alpha}[I]} \boldsymbol{H}_{\tilde\alpha}(\alpha_I)\boldsymbol{G}_{\tilde\alpha}(\alpha_J) \cdot \pi^{(\tilde\alpha)}_{I\cup J}(\alpha_I \oplus \alpha_J). \right), \\ &= \mathop{\mathbb{E}}_{\tilde\alpha \sim \pi_{[n]\setminus(I\cup J)}} \left\langle \boldsymbol{H}_{\tilde\alpha}, \mathsf{C}_{\tilde\alpha}^{J\to I} \boldsymbol{G}_{\tilde\alpha} \right\rangle_{\pi_I^{\tilde\alpha}} \end{aligned} \tag{21}$$

where we have used Equation (7) to get the equality marked by ($\blacklozenge$). To conclude, we observe that both $\boldsymbol{G}_{\tilde\alpha}$ and $\boldsymbol{H}_{\tilde\alpha}$ are perpendicular to the constant functions.

▷ Claim 33. For all $\tilde\alpha \in X[[n] \setminus (I \cup J)]$ we have,

$$\langle \boldsymbol{H}_{\tilde\alpha}, \mathbf{1}\rangle_{\pi_I^{(\tilde\alpha)}} = \langle \boldsymbol{G}_{\tilde\alpha}, \mathbf{1}\rangle_{\pi_J^{(\tilde\alpha)}} = 0.$$

▷ Claim 34. For any $S, T \subset [n]$ with $S \cap T = \varnothing$ and $\boldsymbol{F} : X[S \cup T] \to \mathbb{R}$, we have

$$\|\boldsymbol{F}\|_{S\cup T}^2 = \mathop{\mathbb{E}}_{\alpha \sim \pi_T} \|\boldsymbol{F}_\alpha\|^2_{\pi_S^{(\alpha)}}.$$

[11] in particular, $\pi(\alpha \wedge \bar\alpha) = 0$

Thus, we can continue upper-bounding Equation (21) as follows,

$$\begin{aligned}\langle \boldsymbol{h}, \boldsymbol{g}\rangle_\pi &\le \mathop{\mathbb{E}}_{\tilde\alpha\sim\pi_{[n]\setminus(I\cup J)}}\left[\sigma_2\left(\mathsf{C}_{\tilde\alpha}^{J\to I}\right)\cdot\|\boldsymbol{H}_{\tilde\alpha}\|_{\pi_I^{(\tilde\alpha)}}\cdot\|\boldsymbol{G}_{\tilde\alpha}\|_{\pi_J^{(\tilde\alpha)}}\right],\\ &\le \mathop{\mathbb{E}}_{\tilde\alpha\sim\pi_{[n]\setminus(I\cup J)}}\left[\sigma_2\left(\mathsf{C}_{\tilde\alpha}^{J\to I}\right)\cdot\frac{\|\boldsymbol{H}_{\tilde\alpha}\|^2_{\pi_I^{(\tilde\alpha)}}+\|\boldsymbol{G}_{\tilde\alpha}\|^2_{\pi_I^{(\tilde\alpha)}}}{2}\right],\\ &\le \varepsilon^{J\to I}.\end{aligned}$$

We now present the proofs of Claim 33 and Claim 34.

Proof of Claim 33. Let $\tilde\alpha\in X[[n]\setminus(I\cup J)]$ be fixed. Since the proofs will be identical we will only prove the statement for $\boldsymbol{H}_{\tilde\alpha}$. Since $\boldsymbol{h}$ is picked as an optimizer to Equation (20), we have $\langle \boldsymbol{h}, \boldsymbol{u}_{\tilde\alpha}\rangle_\pi = 0$. We also have,

$$\langle \boldsymbol{u}_{\tilde\alpha}, \boldsymbol{h}\rangle_\pi = \sum_{\alpha\in X[[n]\setminus J]}\boldsymbol{H}(\alpha)\cdot\langle \boldsymbol{u}_{\tilde\alpha}, \boldsymbol{u}_\alpha\rangle_\pi = \sum_{\alpha\in X[[n]\setminus J]}\boldsymbol{H}(\alpha)\cdot\langle \mathbf{1}, \boldsymbol{u}_\alpha\rangle_\pi\cdot\mathbf{1}[\alpha\supset\tilde\alpha],$$

where we have used that $\mathrm{supp}(\boldsymbol{u}_\alpha)\subset\mathrm{supp}(\boldsymbol{u}_{\tilde\alpha})$ whenever $\tilde\alpha\subset\alpha$. Thus, using $\langle \mathbf{1}, \boldsymbol{u}_\alpha\rangle_\pi = \pi_{[n]\setminus J}(\alpha)$ we can simplify the above into

$$0 = \sum_{\substack{\alpha\in X[[n]\setminus J],\\ \alpha\supset\tilde\alpha}}\boldsymbol{H}(\alpha)\cdot\pi_{[n]\setminus J}(\alpha).$$

Every $\alpha\in X[[n]\setminus J]$ satisfying $\alpha\supset\tilde\alpha$ can be written as $\alpha=\tilde\alpha\oplus\alpha_I$ for a unique choice of $\alpha_I\in X_{\tilde\alpha}[I]$. Thus, using $\boldsymbol{H}(\tilde\alpha\oplus\alpha_I)=\boldsymbol{H}_{\tilde\alpha}(\alpha_I)$, we obtain

$$0 = \sum_{\alpha_I\in X_{\tilde\alpha}[I]}\boldsymbol{H}_{\tilde\alpha}(\alpha_I)\cdot\pi_{[n]\setminus J}(\tilde\alpha\oplus\alpha_I) = \pi_{[n]\setminus(I\cup J)}(\tilde\alpha)\cdot\sum_{\alpha_I\in X_{\tilde\alpha}[I]}\boldsymbol{H}_{\tilde\alpha}(\alpha_I)\cdot\pi_I^{(\tilde\alpha)}(\alpha_I),$$

where the last inequality is through Equation (7). The theorem then follows by noting that the term on the RHS is a multiple of $\langle \boldsymbol{H}_{\tilde\alpha}, \mathbf{1}\rangle_{\pi_I^{(\tilde\alpha)}}$. ◁

Proof Claim 34. We have,

$$\begin{aligned}\|\boldsymbol{F}\|^2_{\pi_{S\cup T}} &= \sum_{\alpha\in X[S\cup T]}\boldsymbol{F}(\alpha)^2\cdot\pi_{S\cup T}(\alpha),\\ &= \sum_{\alpha\in X[S\cup T]}\pi_S(\alpha_S)\pi_T^{(\alpha_S)}(\alpha_T)\boldsymbol{F}_{\alpha_S}(\alpha_T)^2,\\ &= \mathop{\mathbb{E}}_{\alpha_S\sim\pi_S}\|\boldsymbol{F}_{\alpha_S}\|^2_{\pi_T^{(\alpha_S)}}\end{aligned}$$

where we have used Equation (7) to obtain the second equality. ◁

◀

References

1. David Aldous. Random walks on finite groups and rapidly mixing Markov chains. In *Séminaire de Probabilités XVII 1981/82*, pages 243–297. Springer, 1983.
2. Vedat Levi Alev, Fernando Granha Jeronimo, Dylan Quintana, Shashank Srivastava, and Madhur Tulsiani. List decoding of lifted codes. In *SODA*, 2020.
3. Vedat Levi Alev, Fernando Granha Jeronimo, and Madhur Tulsiani. Approximating constraint satisfaction problems on high dimensional expanders. In *FOCS*, 2019.

4 Vedat Levi Alev and Lap Chi Lau. Improved analysis of higher order random walks and applications. In *STOC*, pages 1198–1211, 2020. doi:10.1145/3357713.3384317.

5 Vedat Levi Alev and Shravas Rao. Expanderizing higher order random walks, 2024. doi:10.48550/arXiv.2405.08927.

6 Yali Amit. On rates of convergence of stochastic relaxation for gaussian and non-gaussian distributions. *Journal of Multivariate Analysis*, 38(1):82–99, 1991.

7 Nima Anari, Vishesh Jain, Frederic Koehler, Huy Tuan Pham, and Thuy-Duong Vuong. Entropic independence ii: optimal sampling and concentration via restricted modified log-sobolev inequalities. *arXiv preprint*, 2021. arXiv:2111.03247.

8 Nima Anari, Vishesh Jain, Frederic Koehler, Huy Tuan Pham, and Thuy-Duong Vuong. Entropic independence: optimal mixing of down-up random walks. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1418–1430, 2022. doi:10.1145/3519935.3520048.

9 Nima Anari, Kuikui Liu, and Shayan Oveis Gharan. Spectral independence in high-dimensional expanders and applications to the hardcore model. *CoRR*, abs/2001.00303, 2020. arXiv:2001.00303.

10 Mitali Bafna, Max Hopkins, Tali Kaufman, and Shachar Lovett. High dimensional expanders: Eigenstripping, pseudorandomness, and unique games. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1069–1128. SIAM, 2022. doi:10.1137/1.9781611977073.47.

11 Mitali Bafna, Max Hopkins, Tali Kaufman, and Shachar Lovett. Hypercontractivity on high dimensional expanders. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 185–194, 2022. doi:10.1145/3519935.3520040.

12 Antonio Blanca, Pietro Caputo, Zongchen Chen, Daniel Parisi, Daniel Štefankovič, and Eric Vigoda. On mixing of markov chains: Coupling, spectral independence, and entropy factorization. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3670–3692. SIAM, 2022. doi:10.1137/1.9781611977073.145.

13 Antonio Blanca, Pietro Caputo, Daniel Parisi, Alistair Sinclair, and Eric Vigoda. Entropy decay in the swendsen–wang dynamics on zd. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1551–1564, 2021. doi:10.1145/3406325.3451095.

14 Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration Inequalities - A Nonasymptotic Theory of Independence*. Oxford University Press, 2013. doi:10.1093/ACPROF:OSO/9780199535255.001.0001.

15 Russ Bubley and Martin E. Dyer. Path coupling: A technique for proving rapid mixing in Markov chains. In *FOCS*, pages 223–231, 1997. doi:10.1109/SFCS.1997.646111.

16 Djalil Chafaï. Entropies, convexity, and functional inequalities, on ϕ-entropies and ϕ-sobolev inequalities. *Journal of Mathematics of Kyoto University*, 44(2):325–363, 2004.

17 Zongchen Chen, Andreas Galanis, Daniel Stefankovic, and Eric Vigoda. Rapid mixing for colorings via spectral independence. *CoRR*, abs/2007.08058, 2020. arXiv:2007.08058.

18 Zongchen Chen, Kuikui Liu, and Eric Vigoda. Rapid mixing of Glauber dynamics up to uniqueness via contraction. *arXiv preprint*, 2020. arXiv:2004.09083.

19 Mary Cryan, Heng Guo, and Giorgos Mousa. Modified log-Sobolev inequalities for strongly log-concave distributions. *CoRR*, abs/1903.06081, 2019. arXiv:1903.06081.

20 Persi Diaconis and Arun Ram. Analysis of systematic scan metropolis algorithms using iwahori-hecke algebra techniques. *Michigan Mathematical Journal*, 48(1):157–190, 2000.

21 Yotam Dikstein and Irit Dinur. Agreement testing theorems on layered set systems. In *Proceedings of the 60th annual IEEE symposium on foundations of computer science, FOCS 2019, Baltimore, MD, USA, November 10–12, 2019*, pages 1495–1524. Los Alamitos, CA: IEEE Computer Society, 2019. doi:10.1109/FOCS.2019.00088.

22 Yotam Dikstein, Irit Dinur, Yuval Filmus, and Prahladh Harsha. Boolean function analysis on high-dimensional expanders. In *APPROX/RANDOM*, pages 38:1–38:20, 2018. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.38.

23 Irit Dinur and Tali Kaufman. High dimensional expanders imply agreement expanders. In *FOCS*, pages 974–985, 2017. doi:10.1109/FOCS.2017.94.

24 Roland L Dobrushin. Prescribing a system of random variables by conditional distributions. *Theory of Probability & Its Applications*, 15(3):458–486, 1970.

25 Martin Dyer, Leslie Ann Goldberg, and Mark Jerrum. Systematic scan for sampling colorings. *The Annals of Applied Probability*, 16(1):185–230, 2006.

26 Martin Dyer, Leslie Ann Goldberg, and Mark Jerrum. Dobrushin conditions and systematic scan. *Combinatorics, Probability and Computing*, 17(6):761–779, 2008. doi:10.1017/S0963548308009437.

27 Jan Dymara and Tadeusz Januszkiewicz. New kazhdan groups. *Geometriae Dedicata*, 80:311–317, 2000.

28 Weiming Feng, Heng Guo, Yitong Yin, and Chihao Zhang. Rapid mixing from spectral independence beyond the boolean domain. *arXiv preprint*, 2020. arXiv:2007.08091.

29 Jason Gaitonde and Elchanan Mossel. Comparison theorems for the mixing times of systematic and random scan dynamics. *arXiv preprint*, 2024. doi:10.48550/arXiv.2410.11136.

30 Howard Garland. p-adic curvature and the cohomology of discrete subgroups of p-adic groups. *Annals of Mathematics*, pages 375–423, 1973.

31 Zohar Grinbaum-Reizis and Izhar Oppenheim. Curvature criterion for vanishing of group cohomology. *Groups, Geometry, and Dynamics*, 16(3):843–862, 2022.

32 Heng Guo, Kaan Kara, and Ce Zhang. Layerwise systematic scan: Deep boltzmann machines and beyond. In *International Conference on Artificial Intelligence and Statistics*, pages 178–187. PMLR, 2018. URL: http://proceedings.mlr.press/v84/guo18a.html.

33 Heng Guo and Giorgos Mousa. Local-to-global contraction in simplicial complexes, 2021. arXiv:2012.14317.

34 Tom Gur, Noam Lifshitz, and Siqi Liu. Hypercontractivity on high dimensional expanders. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 176–184, 2022. doi:10.1145/3519935.3520004.

35 Thomas P Hayes. A simple condition implying rapid mixing of single-site dynamics on spin systems. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 39–46. IEEE, 2006. doi:10.1109/FOCS.2006.6.

36 Bryan D He, Christopher M De Sa, Ioannis Mitliagkas, and Christopher R é. Scan order in gibbs sampling: Models in which it matters and bounds on how much. *Advances in neural information processing systems*, 29, 2016.

37 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.

38 Fernando Granha Jeronimo, Shashank Srivastava, and Madhur Tulsiani. Near-linear time decoding of ta-shma's codes via splittable regularity. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1527–1536, 2021. doi:10.1145/3406325.3451126.

39 Martin Kassabov. Subspace arrangements and property t. *Groups, Geometry, and Dynamics*, 5(2):445–477, 2011.

40 Tali Kaufman and David Mass. High dimensional random walks and colorful expansion. In *ITCS*, pages 4:1–4:27, 2017. doi:10.4230/LIPIcs.ITCS.2017.4.

41 Tali Kaufman and Izhar Oppenheim. Construction of new local spectral high dimensional expanders. In *STOC*, pages 773–786, 2018. doi:10.1145/3188745.3188782.

42 Tali Kaufman and Izhar Oppenheim. High order random walks: Beyond spectral gap. In *APPROX/RANDOM*, pages 47:1–47:17, 2018. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.47.

43 David A Kazhdan. On the connection of the dual space of a group with the structure of its closed subgroups. *Funkcional. Anal. i Prilozen*, 1(1):71–74, 1967.

44 David A Levin, Yuval Peres, and Elizabeth L Wilmer. Markov chains and mixing times. *American Mathematical Soc., Providence*, 2009.

45 Kuikui Liu. From coupling to spectral independence and blackbox comparison with the down-up walk. *arXiv preprint*, 2021. arXiv:2103.11609.

46 Eyal Lubetzky, Alexander Lubotzky, and Ori Parzanchevski. Random walks on ramanujan complexes and digraphs. *Journal of the European Mathematical Society*, 22(11):3441–3466, 2020.

47 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Explicit constructions of Ramanujan complexes of type. *Eur. J. Comb.*, 26(6):965–993, 2005. doi:10.1016/j.ejc.2004.06.007.

48 Ravi Montenegro and Prasad Tetali. Mathematical aspects of mixing times in Markov chains. *Foundations and Trends in Theoretical Computer Science*, 1(3), 2005. doi:10.1561/0400000003.

49 Izhar Oppenheim. Averaged projections, angles between groups and strengthening of banach property (t). *Mathematische Annalen*, 367:623–666, 2017.

50 Izhar Oppenheim. Angle criteria for uniform convergence of averaged projections and cyclic or random products of projections. *Israel Journal of Mathematics*, 223:343–362, 2018.

51 Izhar Oppenheim. Local spectral expansion approach to high dimensional expanders part I: Descent of spectral gaps. *Discrete & Computational Geometry*, 59(2):293–330, 2018. doi:10.1007/s00454-017-9948-x.

52 Izhar Oppenheim. Local spectral expansion approach to high dimensional expanders part ii: Mixing and geometrical overlapping, 2018. arXiv:1803.01320.

53 Izhar Oppenheim. Banach zuk's criterion for partite complexes with application to random groups. *arXiv preprint*, 2021. arXiv:2112.02929.

54 Izhar Oppenheim. Garland's method with banach coefficients. *Analysis & PDE*, 16(3):861–890, 2023.

55 Ori Parzanchevski. Ramanujan graphs and digraphs. *Anal. Geom. Graphs Manifolds*, 461:344, 2020.

56 Maxim Raginsky. Strong data processing inequalities and ϕ-sobolev inequalities for discrete channels. *IEEE Transactions on Information Theory*, 62(6):3355–3389, 2016. doi:10.1109/TIT.2016.2549542.

57 Gareth O Roberts and Jeffrey S Rosenthal. Surprising convergence properties of some simple gibbs samplers under various scans. *International Journal of Statistics and Probability*, 5(1):51–60, 2015.

58 Laurent Saloff-Coste. Lectures on finite Markov chains. In *Lectures on probability theory and statistics*, pages 301–413. Springer, 1997.

59 Kennan T. Smith, Donald C. Solmon, and Sheldon L. Wagner. Practical and mathematical aspects of the problem of reconstructing objects from radiographs. *Bull. Am. Math. Soc.*, 83:1227–1270, 1977. doi:10.1090/S0002-9904-1977-14406-6.

60 Daniel Stefankovic and Eric Vigoda. Lecture notes on spectral independence and bases of a matroid: Local-to-global and trickle-down from a markov chain perspective, 2023. arXiv:2307.13826.

An Elementary Proof of the First LP Bound on the Rate of Binary Codes

Nati Linial
School of Computer Science and Engineering, Hebrew University, Jerusalem, Israel

Elyassaf Loyfer
School of Computer Science and Engineering, Hebrew University, Jerusalem, Israel

Abstract

The asymptotic rate vs. distance problem is a long-standing fundamental problem in coding theory. The best upper bound to date was given in 1977, and has since received numerous proofs and interpretations. Here we provide a new, elementary proof of this bound that is based on counting walks in the Hamming cube.

2012 ACM Subject Classification Theory of computation → Error-correcting codes

Keywords and phrases Coding theory, code bounds, convex optimization, linear progamming

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.48

Category RANDOM

Related Version *Previous Version*: `https://arxiv.org/abs/2303.16619`

Funding *Nati Linial*: Supported in part by a NSF-BSF research grant "Global Geometry of Graphs".

Acknowledgements The authors are thankful to Leonardo Nagami Coregliano, Fernando Granha Jeronimo and Chris Jones for insightful discussions. We also thank Alex Samorodnitsky for helpful advice and inspiration.

1 Introduction

A binary code of length n is a subset $C \subset \mathbb{F}_2^n$. Its distance is the smallest Hamming distance among all pairs of code words, $\mathrm{dist}(C) := \min_{x \neq y \in C} |x + y|$. A fundamental problem in coding theory is to find the largest possible size, $A(n, d)$, of a code of length n and distance d:

$$A(n, d) := \max\{|C| : C \subset \mathbb{F}_2^n,\ \mathrm{dist}(C) \geq d\}$$

The asymptotic version of this question, the *rate vs. distance problem*, seeks the largest possible rate $R(C) := n^{-1} \log_2(|C|)$, where the distance is linear in n:

$$\mathcal{R}(\delta) := \limsup_{n \to \infty} \{R(C) \colon C \subset \mathbb{F}_2^n,\ \mathrm{dist}(C) \geq \lfloor \delta n \rfloor\}$$

It is known that $\mathcal{R}(0) = 1$, and $\mathcal{R}(\delta) = 0$ for all $\delta \geq 1/2$. The exact value of $\mathcal{R}(\delta)$ remains unknown for all $\delta \in (0, 1/2)$.

The best and only lower bound, $\mathcal{R}(\delta) \geq 1 - H(\delta)$, is due to Gilbert [7] and Varshamov [13]. Here and throughout, $H(\cdot)$ is the binary entropy function.

The best upper bound that we have on $\mathcal{R}(\delta)$ for a large range of δ is the *first linear programming (LP) bound*.[1] Due to McEliece, Rodemich, Rumsey and Welch (MRRW, [9]), this bound states that:

[1] MRRW [9] also proved a second LP bound which improves upon the first for $\delta \in (0, 0.273)$. Because this paper focuses exclusively on the first LP bound, we omit the statement of the second.

© Nati Linial and Elyassaf Loyfer;
licensed under Creative Commons License CC-BY 4.0

▶ **Theorem 1** (The first LP bound [9]). *$\mathcal{R}(\delta) \le H(1/2 - \sqrt{\delta(1-\delta)})$ for every $0 \le \delta \le 1/2$.*

These bounds are based on a linear program due to Delsarte [4], the optimum of which is an upper bound on $A(n,d)$. Although this maximum is still unknown, upper bounds on $A(n,d)$ can be derived by passing to the LP dual and constructing a dual feasible solution.

Numerous works have constructed dual feasible solutions [9, 6, 10, 11, 12, 2, 1], yet all lead to the same asymptotic bound. Interestingly, we observe that not only do all these works reach the same bound, but also construct the same dual feasible solution. They differ, however, in proof techniques, and introduce new insight or generalizations to other metric spaces.

Can we achieve a better bound by designing a different solution? This question motivated the current work. It led us to identify the common pattern underlying previous approaches and develop a slightly more general template for dual solutions. Based on this template, we build a new family of solutions which, while different from previous constructions, yields the same asymptotic upper bound stated in Theorem 1.

Although our new construction does not improve upon Theorem 1, it offers potential insights into the nature of the bound. First, exploring alternate constructions may either lead to improved solutions or clarify the inherent limitations of the Delsarte–MRRW framework. Furthermore, while recent LP/SDP hierarchies offer stronger relaxations to the rate-distance problem ([3, 8]), they have struggled to yield concrete bounds due to their complexity. By constructing an explicit dual solution that relies on different machinery than its predecessors, our walk-based combinatorial approach may prove more adaptable to these advanced hierarchies. Second, our proof technique establishes a connection between bounds on binary codes and small sets with large expansion properties, which can possibly lead to mutual benefits. Third, the main part of our proof is a simple counting argument, which allows for a complete and transparent exposition of the bound, with no technical details omitted. Nonetheless, the proof necessarily includes a non-elementary component common to all previous approaches, detailed in Section 3.

Organization of the paper: Section 2 introduces notation, provides necessary background in Fourier analysis, and recalls Delsarte's LP for binary codes. In Section 3, we introduce the general template for dual feasible solutions and demonstrate how previous works can be seen as implementations of this template. In Section 4, we construct the new solutions. In Appendix A, we state and prove some standard results for completeness.

2 Preliminaries

The Hamming cube. As usual, the Hamming cube is the graph whose vertex set is $\mathbb{F}_2^n$, where adjacency means having Hamming distance one. We denote the adjacency matrix of the cube by $A_H \in \{0,1\}^{\mathbb{F}_2^n \times \mathbb{F}_2^n}$. The j-th *level* of the cube, for $0 \le j \le n$, is the set of vertices of Hamming weight j. We denote by $L_j \colon \mathbb{F}_2^n \to \{0,1\}$ the indicator of the j-th level.

Binary codes. A binary code is a subset $C \subset \mathbb{F}_2^n$. The minimal distance of a code, denoted $\mathrm{dist}(C)$, is determined by the minimum Hamming distance between distinct codewords:

$$\mathrm{dist}(C) \coloneqq \min\{|x-y| \colon x,y \in C \wedge x \neq y\}$$

Analysis of Boolean functions. Let $f, g : \mathbb{F}_2^n \to \mathbb{R}$ be real functions on the cube. We define their inner product w.r.t. the uniform measure,

$$\langle f, g \rangle := 2^{-n} \sum_{x \in \mathbb{F}_2^n} f(x)g(x)$$

Convolution between f and g is defined by

$$(f * g)(x) := 2^{-n} \sum_{y \in \mathbb{F}_2^n} f(y)g(x+y)$$

To every $x \in \mathbb{F}_2^n$ there corresponds a *Fourier character* $\chi_x : \mathbb{F}_2^n \to \mathbb{R}$ that is defined via $\chi_x(y) = (-1)^{\langle x,y \rangle}$. The set of characters $\{\chi_x\}_{x \in \mathbb{F}_2^n}$ forms an orthonormal basis of the space of real functions on the Hamming cube. The Fourier transform of f is denoted $\mathcal{F}[f]$ or $\widehat{f}$ in short, and defined by

$$\widehat{f}(x) := \langle \chi_x, f \rangle = 2^{-n} \sum_{y \in \mathbb{F}_2^n} (-1)^{\langle x,y \rangle} f(y)$$

In Fourier space, operators are defined with normalization factor of 1, and we denote them explicitly with a subscript $\mathcal{F}$, e.g.

$$\langle \widehat{f}, \widehat{g} \rangle_{\mathcal{F}} = \sum_{x \in \mathbb{F}_2^n} \widehat{f}(x) \cdot \widehat{g}(x)$$
$$(\widehat{f} *_{\mathcal{F}} \widehat{g})(x) = \sum_{y \in \mathbb{F}_2^n} \widehat{f}(y)\widehat{g}(y+x)$$

The Fourier transform is its own inverse, up to normalization:

$$f(x) = \mathcal{F}^{-1}[\widehat{f}](x) = \langle \chi_x, \widehat{f} \rangle_{\mathcal{F}} = \sum_{y \in \mathbb{F}_2^n} \chi_x(y)\widehat{f}(y)$$

Parseval's theorem states that $\langle f, g \rangle = \langle \widehat{f}, \widehat{g} \rangle_{\mathcal{F}}$.
The convolution theorem states that $\widehat{f * g} = \widehat{f} \cdot \widehat{g}$.

Krawtchouk polynomials. The Krawtchouk polynomials $\{K_0, \ldots, K_n\}$ are a family of polynomials orthogonal with respect to the binomial probability measure,

$$\sum_{i=0}^{n} 2^{-n} \binom{n}{i} K_j(i)K_k(i) = \begin{cases} \binom{n}{j} & j = k \\ 0 & \text{otherwise} \end{cases}$$

The j-th Krawtchouk can be defined by the sum

$$K_j(t) = \sum_{k=0}^{n} (-1)^k \binom{t}{k} \binom{n-t}{j-k}, \qquad t \in \mathbb{R}$$

In particular, $K_0(t) = 1$ and $K_1(t) = n - 2t$.

2.1 Delsarte's LP

Here we recall Delsarte's LP for binary codes. Although we only require the dual LP, we start with the primal as it better suits the exposition. In his original work [4], Delsarte formulated the LP via association schemes, which readily generalizes to other spaces, e.g. fixed-weight codes or spherical codes. For binary codes with the Hamming distance, the LP can be developed completely through Fourier analysis, as we choose to do here.

Delsarte's LP for binary codes can be stated as follows. Inequalities on functions are defined pointwise.

$$\begin{aligned} \max \quad & \sum_{x \in \mathbb{F}_2^n} f(x) \\ \text{s.t.} \quad & f \colon \mathbb{F}_2^n \to \mathbb{R} \\ & f(0) = 1 && \text{(normalization)} \\ & f \geq 0 && \text{(non-negativity)} \\ & \widehat{f} \geq 0 && \text{(Fourier)} \\ & f(x) = 0 \text{ if } 1 \leq |x| \leq d-1 && \text{(distance)} \end{aligned} \tag{1}$$

▶ **Proposition 2.** *The optimum of Delsarte's LP is an upper bound to $A(n,d)$.*

Proof. The proof is by constructing a feasible solution to (1) whose value is $A(n,d)$. Consider a binary code $C \subset \mathbb{F}_2^n$ of length n, minimal distance $\operatorname{dist}(C) \geq d$, and maximum cardinality, $|C| = A(n,d)$. Let $\mathbb{1}_C : \mathbb{F}_2^n \to \mathbb{R}$ be its indicator function, and let $f_C \coloneqq \frac{2^n}{|C|} \mathbb{1}_C * \mathbb{1}_C$. It is not hard to verify that f_C satisfies the normalization and non-negativity constraints. The Fourier constraints are a consequence of the convolution theorem, as $\mathcal{F}[\mathbb{1}_C * \mathbb{1}_C] = \widehat{\mathbb{1}}_C^2 \geq 0$. The distance constraints follow from the distance property of the code, $\operatorname{dist}(C) \geq d$. Finally, the value of this solution is $\sum_{x \in \mathbb{F}_2^n} f_C(x) = |C|$ which is by assumption $A(n,d)$. ◀

Passing to the LP dual,

$$\begin{aligned} \text{min.} \quad & g(0)/\widehat{g}(0) \\ \text{s.t.} \quad & g \colon \mathbb{F}_2^n \to \mathbb{R} \\ & \widehat{g}(0) > 0 && \text{(normalization)} \\ & \widehat{g} \geq 0 && \text{(Fourier)} \\ & g(x) \leq 0 \text{ if } |x| \geq d && \text{(distance)} \end{aligned} \tag{2}$$

Note that to turn the minimization problem (2) into an honest LP we can just require that $\widehat{g}(0) = 1$.

The dual can be derived by standard LP methods. For completeness we include a proof of weak duality.

▶ **Proposition 3.** *If $g : \mathbb{F}_2^n \to \mathbb{R}$ is feasible for (2), then*

$$A(n,d) \leq g(0)/\widehat{g}(0) \tag{3}$$

Proof. Let $g \colon \mathbb{F}_2^n \to \mathbb{R}$ be feasible as in the proposition's statement, and let $f \colon \mathbb{F}_2^n \to \mathbb{R}$ be an optimal solution to (1). By Proposition 2,

$$\widehat{g}(0) \cdot A(n,d) \leq \widehat{g}(0) \cdot \sum_x f(x) = 2^n \widehat{g}(0) \cdot \widehat{f}(0)$$

Since both $\widehat{f}$ and $\widehat{g}$ are non-negative,

$$\widehat{g}(0) \cdot \widehat{f}(0) \leq \sum_{x \in \mathbb{F}_2^n} \widehat{g}(x) \cdot \widehat{f}(x) = \langle \widehat{g}, \widehat{f} \rangle_{\mathcal{F}}$$

Using Parseval's inequality, we have

$$\langle \widehat{g}, \widehat{f} \rangle_{\mathcal{F}} = \langle g, f \rangle = 2^{-n} \sum_x f(x)g(x)$$

If $1 \leq |x| \leq d-1$ then $f(x) = 0$. If $|x| \geq d$ then $f(x) \geq 0$ while $g(x) \leq 0$. Hence,

$$2^{-n} \sum_x f(x)g(x) \leq 2^{-n} f(0)g(0)$$

Recalling that $f(0) = 1$ and $\widehat{g}(0) > 0$, we conclude that $A(n,d) \leq g(0)/\widehat{g}(0)$. ◀

3 A recipe for dual solutions

We next describe a general framework that yields feasible solutions to (2). The idea is to decouple the two families of constraints: first handle the distance constraints and then the Fourier constraints. We then demonstrate how previous works fit into this framework by implementing a dual solution similar to [9, 6, 10, 11, 12, 2, 1]. Finally, we remark shortly on the proof techniques used in prior works.

▶ **Proposition 4.** *Let $\phi, \Gamma : \mathbb{F}_2^n \to \mathbb{R}$ such that*

1. *$\phi(0) > 0$ and $\phi(x) \leq 0$ whenever $|x| \geq d$,*
2. *The nonzero function Γ satisfies*

$$\widehat{\Gamma} \geq 0, \ \widehat{\phi} *_{\mathcal{F}} \widehat{\Gamma} \geq \widehat{\Gamma}$$

Take $g := \phi \cdot \Gamma^2$. Then, g is feasible for (2), and

$$g(0)/\widehat{g}(0) \leq \phi(0) \cdot |\mathrm{supp}(\widehat{\Gamma})|$$

where $\mathrm{supp}(\widehat{\Gamma})$ is the support of $\widehat{\Gamma}$.

Combined with Proposition 3, Proposition 4 gives an upper bound to $A(n,d)$ in terms of $\phi(0)$ and the support size of $\widehat{\Gamma}$.

Note that this method involves only $\widehat{\Gamma}$ with no direct reference to Γ.

Proof. Let ϕ, Γ and g as above. Indeed, such g is non-positive on levels $d, d+1, \ldots, n$ of the cube, and, by the convolution theorem, $\widehat{g} \geq \widehat{\Gamma} *_{\mathcal{F}} \widehat{\Gamma} \geq 0$. In particular, $\widehat{g}(0) \geq \|\widehat{\Gamma}\|_2^2 > 0$. Hence, g is feasible for (2).

In spirit of [10], we derive an upper bound on $A(n,d)$ in terms of the support size of $\widehat{\Gamma}$. Start with an upper bound on $g(0)$:

$$g(0) = \phi(0) \cdot \Gamma^2(0) = \phi(0) \cdot \left(\sum_{x \in \mathbb{F}_2^n} \widehat{\Gamma}(x) \right)^2$$

Let $S \subset \mathbb{F}_2^n$ be the support of $\widehat{\Gamma}$ and let $\mathbb{1}_S$ be its indicator. Then $\sum_{x \in \mathbb{F}_2^n} \widehat{\Gamma}(x) = \langle \widehat{\Gamma}, \mathbb{1}_S \rangle_{\mathcal{F}}$. By Cauchy-Schwarz,

$$\langle \widehat{\Gamma}, \mathbb{1}_S \rangle_{\mathcal{F}} \leq \left(\langle \widehat{\Gamma}, \widehat{\Gamma} \rangle_{\mathcal{F}} \cdot \langle \mathbb{1}_S, \mathbb{1}_S \rangle_{\mathcal{F}} \right)^{1/2} = \left(\langle \widehat{\Gamma}, \widehat{\Gamma} \rangle_{\mathcal{F}} \cdot |\mathrm{supp}(\widehat{\Gamma})| \right)^{1/2}$$

On the other hand,

$$\widehat{g}(0) \geq (\widehat{\Gamma} *_{\mathcal{F}} \widehat{\Gamma})(0) = \sum_{x \in \mathbb{F}_2^n} \widehat{\Gamma}^2(x) = \langle \widehat{\Gamma}, \widehat{\Gamma} \rangle_{\mathcal{F}}$$

Dividing $g(0)/\widehat{g}(0)$, we get the desired result. ◀

3.1 The MRRW solution

The MRRW construction, and subsequent works, can be described by choosing

- $\phi(x) = 2(d - |x|)$, and
- $\widehat{\Gamma} = f_r$, where f_r is the principal eigenfunction of the adjacency matrix $A_{B(r)}$ of the Hamming ball with radius $r = n/2 - \sqrt{d(n-d)}$ (a more detailed definition is given below).

We turn to give the details in a language close to that of [10]. At the end of this section we remark on the different techniques used by previous authors to obtain this solution.

Let $\phi(x) = 2(d - |x|)$. It can be expressed with Krawtchouk polynomials,

$$\phi(x) = 2(d - |x|) = (n - 2|x|) - (n - 2d) = K_1(|x|) - K_1(d)$$

The Fourier transform of the linear Krawtchouk K_1 is the level-1 indicator L_1, (Proposition 8),

$$\widehat{K_1}(|x|) = L_1(x) = \begin{cases} 1 & |x| = 1 \\ 0 & \text{otherwise} \end{cases}$$

By the convolution theorem,

$$\widehat{\phi \cdot \Gamma}(x) = \widehat{\phi} *_{\mathcal{F}} \widehat{\Gamma} = L_1 *_{\mathcal{F}} \widehat{\Gamma} - K_1(d) \cdot \widehat{\Gamma}$$

The operator "$L_1 *_{\mathcal{F}}$" has a nice interpretation: it is equivalent to multiplication by the adjacency matrix A_H of the Hamming cube (see Proposition 9). The conditions on Γ from Proposition 4 can be expressed as

$$\begin{aligned} &\text{minimize} && |\mathrm{supp}(\widehat{\Gamma})| \\ &\text{s.t.} && \Gamma \neq 0, \quad \widehat{\Gamma} \geq 0, \quad A_H \widehat{\Gamma} \geq (1 + K_1(d)) \cdot \widehat{\Gamma} \end{aligned} \tag{4}$$

The problem is now reduced to finding an approximate eigenfunction of A_H with eigenvalue at least $n - 2d + 1$ and small support. The best known choice is to take $\widehat{\Gamma}$ as the principal eigenfunction of the adjacency matrix restricted to the Hamming ball of radius r.

Let $A_{B(r)} \in \{0,1\}^{\mathbb{F}_2^n \times \mathbb{F}_2^n}$ be the adjacency matrix of the Hamming ball of radius $0 \leq r \leq n$,

$$A_{B(r)}(x,y) = \begin{cases} 1 & |x|, |y| \leq r \wedge |x + y| = 1 \\ 0 & \text{otherwise} \end{cases}$$

We now state facts regarding the operator $A_{B(r)}$, without proof. These facts are compiled from [9, 6, 10, 11, 12, 2, 1], where proofs can be found. The largest eigenvalue of $A_{B(r)}$ is equal to the first root of the r-th Krawtchouk polynomial, and its value is $2\sqrt{r(n-r)} + o(n)$. Let f_r be the corresponding eigenfunction. Then f_r is non-negative and supported on $B(r)$, thus its support size is at most $\sum_{j=0}^{r} \binom{n}{r} = 2^{nH(r/n)+o(n)}$.

Choose $\widehat{\Gamma} = f_r$, for some r to be determined later. Since A_H has only non-negative entries, $A_H f_r \geq 2\sqrt{r(n-r)} \cdot f_r$, element-wise. Consequently, the choice $\widehat{\Gamma} = f_r$ satisfies (4) if

$$2\sqrt{r(n-r)} \geq n - 2d + 1 \tag{5}$$

hence

$$A(n,d) \leq \phi(0) \cdot |\mathrm{supp}(f_r)| \leq 2^{nH(r/n)+o(n)}$$

The first LP bound, Theorem 1, is obtained by optimizing r.

In [9], f_r is obtained using Christoffel-Darboux formula. The problem (4) is equivalent to the Faber-Krahn maximizer in [6]. In [10], f_r is obtained directly by solving a recurrence relation. In [2, 1], a function close to f_r is found by spectral and functional tools. In [5], they use the Rayleigh quotient to prove existence of a feasible solution to (4) supported on levels $[r - \sqrt{n}, r]$ of the Boolean hypercube, for r satisfying (5).

4 A new family of solutions to Delsarte's dual LP

In this section we present our proof, which is cast in the pattern described in the previous section.

The main idea is as follows. By choosing a slightly more involved ϕ than in prior work, we can dramatically simplify $\widehat{\Gamma}$, reducing it to the indicator of a set. The sets of interest are small, and have the property that a random walk, started inside the set, has a high probability of returning to it. This is captured in Proposition 5. The symmetry of the Hamming cube suggests the level sets as natural candidates for such subsets. In Proposition 6 we count the walks between levels of the cube. Finally, we derive the first LP bound (Theorem 1) from Proposition 5 and Proposition 6.

A walk of length m on the cube is a sequence of vertices $(x_1, x_2, \ldots, x_m) \in (\mathbb{F}_2^n)^m$, where every two consecutive vertices are neighbors, $|x_i + x_{i+1}| = 1$, for $i = 1, \ldots, m-1$.

▶ **Proposition 5.** *Let $n, d \in \mathbb{N}$ such that $d \leq n/2$. Let $m \in \mathbb{N}$ and let $S \subset \mathbb{F}_2^n$ be a set of vertices such that for any $x \in S$, the number of length-m walks starting at x and ending in S is at least $(n-2d)^m + 1$. Then,*

$$A(n,d) \leq n^m \cdot |S|$$

Proof. We choose ϕ and Γ to satisfy Proposition 4 and derive the bound. Define

$$\phi_m(x) = (n - 2|x|)^m - (n-2d)^m$$

and let $\widehat{\Gamma} = \mathbb{1}_S$, the indicator function of the set S. Observe that

1. $\phi_m(0) > 0$ and $\phi_m(x) \leq 0$ for $|x| \geq d$.
2. $\widehat{\Gamma} \neq 0$, $\widehat{\Gamma} \geq 0$.

It remains to show that $\widehat{\phi}_m *_{\mathcal{F}} \widehat{\Gamma} \geq \widehat{\Gamma}$. The Fourier transform of $(n-2|x|)^m = K_1^m(x)$ is the m-wise convolution of L_1, the level-1 indictor (see Proposition 8),

$$\mathcal{F}[K_1^m] = \widehat{K}_1 *_{\mathcal{F}} \ldots *_{\mathcal{F}} \widehat{K}_1 = L_1 *_{\mathcal{F}} \ldots *_{\mathcal{F}} L_1$$

Convolution with the level-1 indictor is equivalent to multiplying by the adjacency matrix of the Hamming cube A_H (see Proposition 9). Namely,

$$\widehat{\phi}_m *_{\mathcal{F}} \widehat{\Gamma} = A_H^m \cdot \widehat{\Gamma} - (n-2d)^m \widehat{\Gamma}$$

Hence the remaining condition $\widehat{\phi}_m *_{\mathcal{F}} \widehat{\Gamma} \geq \widehat{\Gamma}$ is equivalent to

$$(A_H^m \cdot \mathbb{1}_S)(x) \geq ((n-2d)^m + 1)\mathbb{1}_S(x) \qquad \forall x \in \mathbb{F}_2^n \tag{6}$$

Let $x \in \mathbb{F}_2^n$. The left-hand side counts walks that start at x and end in S,

$$(A_H^m \mathbb{1}_S)(x) = \sum_{y \in \mathbb{F}_2^n} A_H^m(x,y)\mathbb{1}_S(y) = \sum_{y \in S} A_H^m(x,y)$$

the last expression sums the number of length-m paths between x and y, for every $y \in S$.

If $x \notin S$, then the left-hand side of (6) is non-negative while the right-hand side is 0. If $x \in S$, the number of walks from x to S exceeds $((n-2d)^m+1)$, by assumption. Hence (6) holds for every $x \in \mathbb{F}_2^n$, and ϕ_m, Γ satisfy Proposition 4.

Finally, $\phi_m(0) = n^m - (n-2d)^m \leq n^m$, and the support size of $\mathbb{1}_S$ is $|S|$. By Proposition 4, $A(n,d) \leq n^m \cdot |S|$. ◀

▶ **Proposition 6.** *Let $j, r, m \in \mathbb{N}$ such that $m = o(\sqrt{r})$ and $r \leq n/2$. Let $P_{r,m,j}$ be the number of walks that start at some vertex x in level r and end at any vertex in level j. Then,*

1. *If $m + r + j = 1 \pmod 2$ or $m < |r - j|$ then $P_{r,m,j} = 0$.*
2. *Otherwise,*

$$P_{r,m,j} = \binom{m}{(m-r+j)/2}(n-r)^{(m+r-j)/2} r^{(m-r+j)/2}(1+O(m/r))^m$$

Proof. A vertex in level k has $n-k$ neighbors in level $k+1$, and k neighbors at level $k-1$. The walk takes m steps and thus contained in levels $r-m, \ldots, r+m$, so from every vertex encountered during the walk there are $n - r + O(m)$ ways to go up one level and $r + O(m)$ ways to go down one level.

If the walk terminates at level j, it takes in total $(m+r-j)/2$ steps up and $(m-r+j)/2$ steps down. There are $\binom{m}{(m-r+j)/2}$ ways to choose the order of the up/down steps, $n-r+O(m)$ options for each "up" step and $r + O(m)$ options for each "down" step, so in total

$$\binom{m}{(m-r+j)/2}(n-r+O(m))^{(m+r-j)/2}(r+O(m))^{(m-r+j)/2}$$ ◀

▶ **Corollary 7.** *Let $1 \leq d \leq n/2$ be integers. Then $A(n,d) \leq 2^{nH(1/2-\sqrt{d/n(1-d/n)})+o(n)}$.*

Proof. Let m be odd, $m = \omega_n(1)$ and $m = o(\sqrt{n})$. Let $r = n/2 - \sqrt{d(n-d)}$ and let $S = \{r-1 \leq |x| \leq r\}$. It suffices to show that S has sufficiently many returning walks. Let $x \in S$. If $|x| = r$, by Proposition 6, the number of length-m walks from x to level r is 0, and to level $r-1$ it is

$$P_{r,m,r-1} = \binom{m}{(m-1)/2}\sqrt{((n-r)r)^m \frac{n-r}{r}}(1+O(m/r))^m$$

Using the well known approximation for binomial coefficients, $\binom{n}{k} = 2^{nH(k/n)-O(\log n)}$, we obtain

$$((A_H^m \mathbb{1}_S)(x))^{1/m} \geq 2\sqrt{r(n-r)} + O(m)$$

By our choice of r, this is greater than $((n-2d)^m+1)^{1/m}$. The proof for $x \in S$ with $|x| = r-1$ is similar. ◀

Theorem 1 follows directly from the above corollary.

References

1 Alexander Barg and Dmitry Nogin. A functional view of upper bounds on codes. In *Coding and cryptology*, pages 15–24. World Scientific, 2008.

2 Alexander M Barg and D Yu Nogin. Spectral approach to linear programming bounds on codes. *Problems of Information Transmission*, 42(2):77–89, 2006. doi:10.1134/S0032946006020025.

3 Leonardo Nagami Coregliano, Fernando Granha Jeronimo, and Chris Jones. A complete linear programming hierarchy for linear codes. *arXiv preprint arXiv:2112.09221*, 2021. arXiv:2112.09221.

4 Philippe Delsarte. An algebraic approach to the association schemes of coding theory. *Philips Res. Rep. Suppl.*, 10:vi+–97, 1973.

5 Dean Doron and Amnon Ta-Shma. Lecture notes on error correcting codes, 2017. Accessed: 2026-05-04. URL: `https://www.math.tau.ac.il/~amnon/Classes/2017-ECC/Lectures/LPBound.pdf`.

6 Joel Friedman and Jean-Pierre Tillich. Generalized Alon–Boppana theorems and error-correcting codes. *SIAM Journal on Discrete Mathematics*, 19(3):700–718, 2005. `doi:10.1137/S0895480102408353`.

7 Edgar N Gilbert. A comparison of signalling alphabets. *The Bell system technical journal*, 31(3):504–522, 1952.

8 Elyassaf Loyfer and Nati Linial. New lp-based upper bounds in the rate-vs.-distance problem for binary linear codes. *IEEE Transactions on Information Theory*, 69(5):2886–2899, 2023. `doi:10.1109/TIT.2023.3236660`.

9 Robert McEliece, Eugene Rodemich, Howard Rumsey, and Lloyd Welch. New upper bounds on the rate of a code via the Delsarte-Macwilliams inequalities. *IEEE transactions on Information Theory*, 23(2):157–166, 1977. `doi:10.1109/TIT.1977.1055688`.

10 Michael Navon and Alex Samorodnitsky. On Delsarte's linear programming bounds for binary codes. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, pages 327–336. IEEE, 2005.

11 Michael Navon and Alex Samorodnitsky. Linear programming bounds for codes via a covering argument. *Discrete & Computational Geometry*, 41(2):199–207, 2009. `doi:10.1007/S00454-008-9128-0`.

12 Alex Samorodnitsky. One more proof of the first linear programming bound for binary codes and two conjectures. *arXiv preprint arXiv:2104.14587*, 2021. `arXiv:2104.14587`.

13 Rom Rubenovich Varshamov. Estimate of the number of signals in error correcting codes. *Docklady Akad. Nauk, SSSR*, 117:739–741, 1957.

A Supplementary proofs

▶ **Proposition 8.** *Let K_j be the j-th Krawtchouk polynomial,*

$$K_j(t) = \sum_{k=0}^{n} (-1)^k \binom{t}{k}\binom{n-t}{j-k}$$

and let $L_j(x) = \mathbb{1}_{|x|=j}$ be the indicator function of the j-level of the Hamming cube. Then,

$$\widehat{K_j} = L_j$$

Proof. It is easier to apply the Fourier transform to L_j, and recall that Fourier transform is its own inverse up to normalization, $\mathcal{F}^{-1}[\mathcal{F}[f]] = 2^n \mathcal{F}[\mathcal{F}[f]] = f$, for every $f\colon \mathbb{F}_2^n \to \mathbb{R}$.

Indeed, let $x \in \mathbb{F}_2^n$ with $|x| = t$. Then,

$$2^n \widehat{L}_j(x) = \sum_{y\in\mathbb{F}_2^n} L_j(y)\chi_x(y) = \sum_{|y|=j} (-1)^{\langle x,y\rangle}$$

where $\langle x, y\rangle = \sum_{i=1}^n x_i y_i$. We group elements by the value of $\langle x, y\rangle$:

$$\sum_{|y|=j} (-1)^{\langle x,y\rangle} = \sum_{k=0}^{n} \sum_{\substack{y\in\mathbb{F}_2^n:\\ |y|=j\\ \langle y,x\rangle=k}} (-1)^k$$

The term $(-1)^k$ does not depend on y anymore, so it remains to count $|\{y \in \mathbb{F}_2^n \colon |y| = j \langle y, x\rangle = k\}|$. If $\langle x, y\rangle = k$, there are $\binom{|x|}{k} = \binom{t}{k}$ options to choose the bits of y that intersect with x, and $\binom{n-|x|}{|y|-k} = \binom{n-t}{j-k}$ options to choose the bits of y outside the support of x. Hence

$$2^n \widehat{L}_j(x) = \sum_{k=0}^{n} (-1)^k \binom{t}{k} \binom{n-t}{j-k}$$ ◀

▶ **Proposition 9.** *Let $f \colon \mathbb{F}_2^n \to \mathbb{R}$, then $\widehat{K_1} *_{\mathcal{F}} f = A_H \cdot f$, where A_H is the adjacency matrix of the Hamming cube.*

Proof. By Proposition 8, $\widehat{K}_1 *_{\mathcal{F}} f = L_1 *_{\mathcal{F}} f$. By definition,

$$(\widehat{K}_1 *_{\mathcal{F}} f)(x) = (L_1 *_{\mathcal{F}} f)(x) = \sum_{y \in \mathbb{F}_2^n} L_1(x+y) f(y) = \sum_{y \in \mathbb{F}_2^n \colon |x+y|=1} f(y)$$

The condition $|x + y| = 1$ is the adjacency condition between vertices x, y in the Hamming cube, hence the last sum is equal $(A_H f)(x)$. ◀

Ultra-Sparse Expanders and the Free Method

Gil Cohen
Tel Aviv University, Israel

Gal Maor
Tel Aviv University, Israel

Abstract

In this paper we ask how much expansion one can retain with almost no edges beyond connectivity. Concretely, for graphs of average degree $2+\varepsilon$, what is the "Ramanujan bound" – how does spectral expansion scale with ε? We compare five ultra–sparse graph models – including the configuration model, subdivision of regular expanders, and the union of a cycle with a partial matching – and analyze each under the normalized or unnormalized notions of expansion. In the normalized setting, we prove bounds that are essentially optimal, determining the correct asymptotic dependence on ε. Our results extend to expansion in general irregular graphs.

For some models we prove rigorous bounds – primarily via finite free probability – while others remain beyond our current techniques. To bridge this gap, we introduce the *Free Method*, which produces quantitative predictions without proving existence – analogous to the probabilistic method, which certifies existence without providing an explicit construction. These predictions align with experiments. We expect the free method to be useful more broadly in graph-theoretic settings.

2012 ACM Subject Classification Mathematics of computing → Spectra of graphs; Theory of computation → Expander graphs and randomness extractors

Keywords and phrases Spectral expanders, free probability

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.49

Category RANDOM

Related Version *Full Version*: `https://eccc.weizmann.ac.il/report/2025/196/`

Funding Funded by ERC starting grant 949499 and by the Israel Science Foundation grant 2989/24.

1 Introduction

Expanders are sparse, highly connected graphs whose expansion allows them, up to some error, to behave like complete graphs at far lower cost. They have been studied for decades, with constructions ranging from combinatorial [31, 5, 4] and group-theoretic [20, 25] to analytic [22]. Their applications include coding theory [33, 2, 36], computational complexity [17, 13, 30], and more. For background, see [16], Chapter 4 of [37], or [35].

We study the optimal expansion when the edge budget is barely above connectivity. Specifically, for graphs of average degree $2+\varepsilon$, what is the "Ramanujan bound" – how does spectral expansion depend on ε? Such *ultra-sparse* graphs are necessarily irregular, as are many graphs arising in applications, including sparse codes [21, 32]. We therefore consider both normalized and unnormalized (adjacency-based) expansion.

Let G be an undirected graph on n vertices with adjacency matrix A and eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. We call λ_2 the *one-sided spectral expansion*, $\lambda_1 - \lambda_2$ the *one-sided spectral gap*, and $\lambda \stackrel{\triangle}{=} \max\{|\lambda_2|, |\lambda_n|\}$ the (two-sided) *spectral expansion*. In the normalized setting, let D be the diagonal degree matrix and $W \stackrel{\triangle}{=} AD^{-1}$ the random walk matrix, with eigenvalues $1 = \omega_1 \geq \omega_2 \geq \cdots \geq \omega_n$. Here ω_2 is the *one-sided normalized spectral expansion*, $1-\omega_2$ the corresponding gap, and $\omega \stackrel{\triangle}{=} \max\{|\omega_2|, |\omega_n|\}$ the (two-sided) *normalized spectral expansion*.


© Gil Cohen and Gal Maor;
licensed under Creative Commons License CC-BY 4.0

A connected 2-regular graph is a cycle, whose spectral gap vanishes with its size. By contrast, 3-regular graphs can attain normalized spectral expansion $\frac{\sqrt{8}}{3}$, bounded away from 1. More generally, the optimal asymptotic value for d-regular graphs is $\frac{2\sqrt{d-1}}{d}$ [28], attained by Ramanujan graphs. Thus, greater degree buys better expansion.

1.1 Ultra-sparse expanders

Since the main utility of expanders is normalized spectral expansion bounded away from 1, we ask how sparse a graph can be while retaining this property, even beyond the regular-graph barrier $\frac{\sqrt{8}}{3} \approx 0.94$. Our goal is to characterize the sparsity–expansion tradeoff in this regime.

Any connected n-vertex graph has average degree at least $2 - \frac{2}{n}$, so we focus on average degree $2+\varepsilon$, where $\varepsilon > 0$ controls the sparsity. We seek the corresponding Ramanujan bound as a function of ε.

To keep degree variance small, we mainly study graphs with degrees in $\{2, 3\}$. High-degree vertices may impose disproportionate load, so we exclude hub-and-spoke examples such as the wheel graph, even though they can be more amortization-efficient. Our methods extend to broader degree distributions. In the normalized setting we obtain essentially optimal bounds, while the unnormalized setting remains more open.

For irregular graphs, normalized and unnormalized expansion capture different phenomena. The normalized spectrum governs random walks, while the unnormalized spectrum is better suited to edge expansion and expander-mixing-type bounds, with different treatments of degree heterogeneity; see [1]. Although less studied for irregular graphs, the unnormalized spectrum remains informative when the degree variance is small, as in the $\{2, 3\}$ regime.

There are two main frameworks for constructing Ramanujan graphs. The first uses highly symmetric Cayley graphs of carefully chosen non-abelian groups, beginning with Lubotzky, Phillips, and Sarnak [20] and Margulis [25]. The second, initiated by Marcus, Spielman, and Srivastava, is analytic and draws on free probability [22, 23, 24]; Cohen later made this approach explicit [11].

The group-theoretic framework appears poorly suited to ultra-sparse graphs because Cayley graphs are regular. Irregular group-based constructions, for example via Schreier graphs, remain an interesting direction.

The MSS framework studies unions of d suitably uncorrelated perfect matchings and targets one-sided expansion. Although it does not directly match our setting, it is our main point of departure.

1.2 Five models of ultra-sparse graphs

The MSS approach studies a graph distribution and proves that some graph in its support is Ramanujan. For d-regular graphs, the distribution is the union of d perfect matchings. We ask which analogous models are natural in the ultra-sparse regime, and consider five.

Two-and-a-fraction matchings

Our first model is the union of two perfect matchings and an ε-matching, denoted $\mathcal{M}+\mathcal{M}+\varepsilon\mathcal{M}$ or $(2+\varepsilon)\mathcal{M}$. The partial matching covers an ε-fraction of the vertices. At $\varepsilon = 0$ and $\varepsilon = 1$, this recovers the MSS models for 2- and 3-regular graphs.

Three equally sized partial matchings

Alternatively, take three β-matchings with $\beta = \frac{2+\varepsilon}{3}$, denoted $3\left(\frac{2+\varepsilon}{3}\mathcal{M}\right)$. This model has a basic connectivity drawback: for small ε, the expected fraction of isolated vertices is $\left(\frac{1-\varepsilon}{3}\right)^3 \approx \frac{1}{27}$. We nevertheless study it to compare the models rather than select a single "best" one.

Cycle plus partial matching

To guarantee connectivity, we also study the union of a spanning cycle and an ε-partial matching, denoted $\mathcal{C} + \varepsilon\mathcal{M}$. Unlike $(2+\varepsilon)\mathcal{M}$, this model is connected before the partial matching is added.

A related explicit 3-regular construction [19] joins each field element x to $x+1$, $x-1$, and x^{-1}, forming a cycle plus a matching. A similar model was recently studied from a property-testing perspective [14].

The configuration model

Here each vertex v contributes d_v half-edges, and a uniformly chosen perfect matching on all half-edges determines the graph. In our ultra-sparse setting, $d_v \in \{2, 3\}$.

The configuration model is standard in spectral random graph theory [6] and in the analysis of graph operators such as the zig-zag product [31, 9]. Unlike a union of perfect matchings, it naturally supports varying degrees. Our analysis also gives another proof of the MSS result in the regular case.

Subdividing 3-regular graphs

Finally, start from a 3-regular Ramanujan graph, or more generally a d-regular one, and replace every edge by a long path. This subdivision operation, classical since Kuratowski [18], inserts degree-2 vertices and drives the average degree toward 2. Choosing the path length appropriately yields average degree $2+\varepsilon$ while preserving the base graph's global structure.

This model is explicit whenever the base graph is explicit, and strongly explicit whenever the base graph is strongly explicit.

2 Our Results

In this section, we present our results on the five ultra-sparse graph models introduced above. In the normalized setting, we essentially settle the question of the optimal ultra-sparse expanders. In the unnormalized setting, we introduce a new framework for analyzing the problem, while many basic questions remain open.

As an additional application, the full version proves that suitable unions of Ramanujan graphs remain nearly Ramanujan, and that suitable unions of cycles attain the Ramanujan bound.

2.1 Ultra-sparse expanders in the normalized setting

We begin by presenting our results in the normalized setting. Our first result uses the configuration model (the fourth model) and applies to general irregular graphs – indeed, to any prescribed degree sequence.

▶ **Theorem 1** (Irregular graphs via the configuration model). *Let $(d_1, \ldots, d_n)$ be a sequence of degrees such that $\sum_{i=1}^{n} d_i$ is even. Then there exists a graph with this degree sequence whose one-sided normalized spectral expansion is bounded above by $\frac{2\sqrt{\bar{d}-1}}{\bar{d}}$, where $\bar{d}$ is the average degree.*

Our proof is based on the configuration model (rather than the perfect matchings model used in [23]) and is inspired by the notion of *free projections* from classical free probability theory, which we discuss in Section 5. As a direct corollary we obtain the following result for ultra-sparse graphs.

▶ **Corollary 2** (Ultra-sparse expanders via the configuration model). *For every $\varepsilon > 0$ and for infinitely many values of n, there exists a graph on n vertices with degrees 2 or 3, with average degree $2 + \varepsilon$, whose one-sided normalized spectral expansion is bounded by $\frac{2\sqrt{1+\varepsilon}}{2+\varepsilon} = 1 - \frac{\varepsilon^2}{8} + O(\varepsilon^3)$.*

At first glance, the bound in Theorem 1 resembles the Ramanujan bound for d-regular graphs. Moreover, empirical evidence suggests that the typical behavior of random configuration graphs is indeed captured by Theorem 1. However, optimality in the irregular setting is more delicate, and the bound is not generally tight. For example, consider the wheel graph, formed from a cycle plus one additional vertex connected to all cycle vertices (we elaborate on this example in the full version of this paper). The normalized spectral expansion of the wheel approaches $\frac{2}{3}$, since within two steps one reaches a uniformly random vertex with probability $\frac{1}{3}$. Yet its average degree approaches 4 as the number of vertices grows, so Theorem 1 guarantees only a graph G with $\omega(G) \leq \frac{\sqrt{3}}{2} \approx 0.866$, which is significantly worse.

This gap suggests that even for graphs with degrees 2 and 3 – where degree variance is kept to a minimum – Corollary 2 may not be tight. Indeed, our second result shows that the subdivision model (our fifth model) does yield better expansion. For a d-regular graph G and parameter $\varepsilon > 0$, let G_ε denote the graph obtained by subdividing each edge of G into paths of equal length, so that the resulting graph has average degree $2 + \varepsilon$.

▶ **Theorem 3** (Ultra-sparse expanders via subdivision). *Let G be a d-regular graph with normalized spectral expansion ω, and let $\varepsilon > 0$. Then the normalized spectral expansion of G_ε is $\cos\left(\frac{d}{2(d-2)} \arccos(\omega)\varepsilon\right)$.*

Specializing to $d = 3$ and taking G be a 3-regular Ramanujan graph, the guaranteed spectral expansion for the resulting ultra-sparse graph is the following.

▶ **Corollary 4.** *Let G to be a 3-regular Ramanujan graph. The normalized spectral expansion of G_ε is $\cos\left(\frac{3}{2}\arccos\left(\frac{\sqrt{8}}{3}\right)\varepsilon\right) \approx 1 - 0.1299\varepsilon^2$.*

This improves upon the bound obtained via the configuration model in Corollary 2. In particular, the syntactically Ramanujan-like bound in Theorem 1 is not optimal.

It is, perhaps, surprising that the configuration model yields weaker expanders. In the configuration model, the degree-2 vertices are incorporated *while* the expander is being constructed, whereas in the subdivision approach they are added only after the 3-regular Ramanujan base graph has been fixed – and in an indifferent, uniform manner: every edge is replaced by a path of the same length. However, the Ramanujan property only guarantees a large spectral gap; it conveys little about local edge densities, so some regions can still be denser than others. Thus, it can be advantageous to distribute the degree-2 vertices nonuniformly, taking into account these density variances.

On the other hand, the expected time spent "stuck" on a path – which weakens the spectral expansion – is quadratic in its length. Since the total length summed over all paths is fixed as a function of ε, it is therefore beneficial to make all paths the same length. Still, one might hope to exploit a 3-regular Ramanujan base graph with substantial variation in local edge density by distributing the subdivision lengths nonuniformly, potentially yielding an even larger spectral gap.

The gap is, however, bounded, in a concrete way that shows both Corollaries 2 and 4 are asymptotically optimal in the dependence on ε. The aforementioned quadratic time to escape a path is levaraged to obtain the following lower bound (proved in the full version), serving as an asymptotic "Alon-Boppana" bound for ultra-sparse graphs.

▶ **Theorem 5.** *Let G be a graph with all vertices of degrees 2 or 3 and average degree $2+\varepsilon$. Then the normalized spectral expansion of G is at least $1-O(\varepsilon^2)$.*

This leads to the following open problem.

▶ **Open Problem 6.** *What is the optimal normalized spectral gap for graphs with degrees 2 and 3 and average degree $2+\varepsilon$?*

Corollary 4, together with Theorem 5, prove that the true answer to Open Problem 6 is $C\varepsilon^2 + o\left(\varepsilon^2\right)$ for some constant $C \geq 0.1299$.

2.2 Ultra-sparse expanders in the unnormalized setting

In this section we present our results in the unnormalized setting. We begin with the subdivision model, presenting our result for ultra-sparse graphs in the unnormalized setting, and then turn to the other four models in the same setting. Although we are not yet able to prove the existence of ultra-sparse graphs for these other models in the unnormalized setting, we introduce the *Free Method*, which yields quantitative *predictions* for the attainable spectral expansion as a function of ε in each case. While nonrigorous, these predictions align with empirical observations. We view the Free Method, discussed in Section 6, as a main contribution alongside our formal results: it lets us "peek" at the expected behavior before an existence proof is in hand.

▶ **Theorem 7** (Ultra-sparse expanders via subdivision)**.** *Let G be a d-regular graph with spectral expansion λ, and let $\varepsilon > 0$. Then the spectral expansion of G_ε is obtained as follows: find the unique positive ξ solving*

$$\lambda = \frac{\sinh((k+1)\xi) - (d-1)\sinh((k-1)\xi)}{\sinh(\xi)}. \tag{1}$$

where $k = \left\lceil \frac{(d-2)(2+\varepsilon)}{d\varepsilon} \right\rceil$. Then the spectral expansion μ is given by $2\cosh\xi$.

Although Theorem 7 gives the exact one-sided spectral expansion, it is not straightforward to read off its dependence on ε (a numerical derivation for μ for the case where G is a 3-regular Ramanujan graph is shown in Figure 1). However, by expanding about $\varepsilon = 0^+$, we show that $\mu = \sqrt{d-1} + \frac{1}{\sqrt{d-1}} + (1+o_k(1))\frac{\lambda}{2}\cdot\frac{(d-2)^2}{\left(\sqrt{d-1}\right)^{k+3}}$. Observe the *exponential* decay of the error in $k \approx 1/\varepsilon$, in contrast to the normalized setting, where the decay in ε is only quadratic – even for the same model.

We now switch gears to the other four models and the free method we use to analyze them.

2.3 The free method

We view the free method as an analogue of the probabilistic method, and so we begin by discussing the latter. The probabilistic method is a powerful tool: it proves the *existence* of an object without revealing what it looks like. The probabilistic method offers no insight into the structure of the object or how to find it explicitly. The value associated with the guaranteed object often serves as a benchmark (a quantitative prediction) for researchers aiming to find explicit constructions – e.g., the dependence of k on n for Ramsey graphs, or the distance–rate tradeoff in coding theory.

However, in many graph-theoretic settings even proving existence is technically formidable. Consider, for example, the value $2\sqrt{d-1}$. Proving the existence of graphs that attain this spectral expansion is notoriously difficult. Nonetheless, the value itself – serving as a benchmark – is both informative and insightful.

The free method, in a sense, takes things one step further. It is designed for situations where even the *mere existence* of an object with the desired properties is too difficult to prove, and thus the optimal value cannot be established via the same route as in the probabilistic method (namely, by proving existence). We reiterate that the main takeaway from the probabilistic method is often the *value* that serves as a benchmark for explicit constructions. When it comes to devising such constructions, the existence proof is typically unused, as it generally ignores precisely the structure required for explicit constructions. The free method provides *only* this key information.

More concretely, the free method targets graph-theoretic problems by abstracting away everything except the spectrum. Given a problem on finite graphs, we first *model* it by an infinite-dimensional object that captures the essential spectral features of the original problem. We then apply tools from free probability (powerful yet simple to use) to solve the infinite-dimensional problem, and the resulting quantity serves as a prediction for the finite case. We denote such predicted values by $\mathsf{pred}(\cdot)$ throughout the paper.

This perspective generalizes the "universal cover" viewpoint, but without requiring identification of a specific cover or combinatorial structure – only the spectral data.

To summarize, the philosophy of the probabilistic method is to prove existence without providing an explicit construction, while also yielding a quantitative benchmark – often the most valuable piece of information when constructing explicit objects. In contrast, the free method aims to derive this benchmark while entirely sidestepping a proof of existence, let alone an explicit construction. Because it makes weaker guarantees, the free method applies in much more challenging settings – ubiquitous in spectral graph theory – where the probabilistic method often falls short due to intricate eigenvalue behavior and the problem's inherent noncommutativity.

To set the stage for a more formal discussion, we fix notation and recall the *Cauchy transform* of an undirected graph G on n vertices: $\mathcal{G}_G(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{x-\lambda_i}$.

Consider the $3\left(\frac{2+\varepsilon}{3}\mathcal{M}\right)$ model, the union of three ρ-partial matchings with $\rho = \frac{2+\varepsilon}{3}$. In settings where one unions t independent samples from the same distribution (here $t = 3$), the first procedure we present in the free method prescribes:

1. Find the unique solution $x_0 > \lambda_1$ to

$$(t-1)\,\mathcal{G}'_G(x) + t\,\mathcal{G}_G(x)^2 \;=\; 0. \tag{2}$$

2. The prediction is given by

$$t \cdot x_0 \;-\; \frac{t-1}{\mathcal{G}_G(x_0)}. \tag{3}$$

At this point we have not justified this procedure (let alone the existence and uniqueness of $x_0 > \lambda_1$). The point is that the resulting procedure is quite general and easy to apply. In Section 6 we explain the rationale and derivation; here we focus solely on applying the method.

Before applying the free method to ultra-sparse graphs, we warm up by rederiving the Ramanujan bound $2\sqrt{d-1}$ using the recipe above.

2.3.1 Predictions for the three partial matchings model via the free method

We now apply the free method to ultra-sparse expanders, in particular to the $3\left(\frac{2+\varepsilon}{3}\mathcal{M}\right)$ model. Concretely, we take $t = 3$ independent copies of a ρ-partial matching with $\rho = \frac{2+\varepsilon}{3}$. Such a matching has a $(1-\rho)$-fraction of its eigenvalues equal to 0, with the remaining eigenvalues split evenly between $+1$ and -1. The corresponding Cauchy transform, denoted $\mathcal{G}_\rho$, is therefore $\mathcal{G}_\rho(x) = \frac{x^2-1+\rho}{x(x^2-1)}$. Equation (2) now becomes $2\,\mathcal{G}'_\rho(x) + 3\,\mathcal{G}_\rho(x)^2 = 0$, whose unique solution for $x > 1$ is

$$x_0(\rho) = \sqrt{1 + \sqrt{\rho(4-3\rho)}}\,.$$

Plugging this to Equation (3) and expressing the result in terms of ε yields the prediction

$$\begin{aligned}\mathsf{pred}_{3\left(\frac{2+\varepsilon}{3}\mathcal{M}\right)}(\varepsilon) &= \sqrt{1 + \sqrt{\frac{4-\varepsilon^2}{3}} \cdot \frac{\sqrt{3(4-\varepsilon^2)} - 3\varepsilon}{2(1-\varepsilon)}} \\ &= \sqrt{3+\sqrt{12}} + \frac{\sqrt{3+\sqrt{12}}}{4+\sqrt{12}}\,\varepsilon + O(\varepsilon^2).\end{aligned} \tag{4}$$

On the other extreme, when $\varepsilon = 1$, we recover the $2\sqrt{2}$ bound for 3-regular Ramanujan graphs. In fact, the Taylor expansion to the left of $\varepsilon = 1$ is $2\sqrt{2} - \frac{1-\varepsilon}{3\sqrt{2}} + O\left((1-\varepsilon)^2\right)$. This ε–dependence is hard to infer purely from experiments or by combinatorial means alone, and it matches experimental results.

2.3.2 Predictions for the remaining models

In Section 6–where we also explain the rationale behind the method–we apply the free method to the remaining models using techniques analogous to the procedure introduced above. For the configuration model, we apply a procedure to bound the spectrum of the product of two matrices, one of which is PSD, which has common ground with the proof of Theorem 1. Its application gives

$$\mathsf{pred}_{\mathsf{conf}}(\varepsilon) = \frac{3}{\sqrt{2}} + \frac{3}{4}\,\varepsilon^{1/2} + \frac{9\sqrt{2}}{32}\,\varepsilon + O(\varepsilon^{3/2}). \tag{5}$$

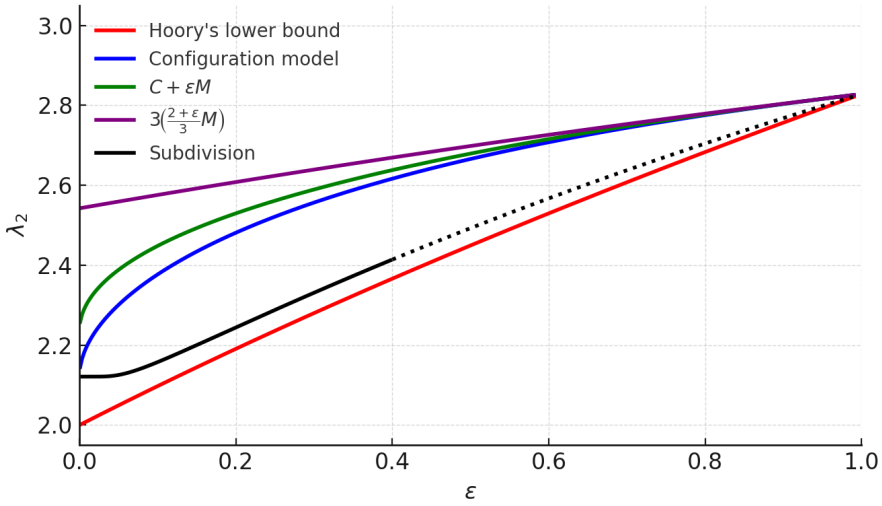


Figure 1 Comparison of the predictions for the bound on λ_2 for the conf, $\mathcal{C} + \varepsilon\mathcal{M}$ and $3\left(\frac{2+\varepsilon}{3}\mathcal{M}\right)$ models. Hoory's lower bound of $2\sqrt{\bar{d}-1}$ is displayed in red, and the subdivision model of a 3-regular Ramanujan graph ($\lambda = 2\sqrt{2}$) appears in black. Note that, as the number of subdivisions k is an integer, the largest possible value for ε in this model is 0.4. For clarity, however, we also plot the solution to Equation (1) for real $k < 2$ as a dotted line.

Interestingly, although $(2+\varepsilon)\mathcal{M}$ and $\mathcal{C} + \varepsilon\mathcal{M}$ are quite different as finite-graph models, their behavior coincides under the free-method analysis (based on classical free probability). Using an additive analog of the multiplicative procedure used for the configuration model yields the prediction

$$\mathsf{pred}_{\mathcal{C}+\varepsilon\mathcal{M}}(\varepsilon) = \mathsf{pred}_{(2+\varepsilon)\mathcal{M}}(\varepsilon) = \sqrt{5} + \sqrt{\frac{4}{5\phi}}\,\sqrt{\varepsilon}, \tag{6}$$

where $\phi = (1+\sqrt{5})/2$ is the golden ratio. The analysis uses a variant of the procedure used for $3\left(\frac{2+\varepsilon}{3}\mathcal{M}\right)$, applied to two distinct distributions.

For comparison, Hoory's lower bound [15] states that if a graph G has average degree at least $\bar{d}$ even after deleting any ball of radius r, then

$$\lambda_2(G) \;\geq\; 2\sqrt{\bar{d}-1}\left(1 - c\,\frac{\log r}{r}\right), \tag{7}$$

for a universal constant c. In our unbounded-size models the r-dependent error should be negligible, so we compare to $2\sqrt{\bar{d}-1} = 2\sqrt{1+\varepsilon}$. Together with our predictions, this yields the comparison shown in Figure 1. As the figure suggests, there is an ordering of the models, and the curves coincide only at $\varepsilon = 1$.

3 Ultra-Sparse Expanders via Subdivision

Let G be a d-regular graph and denote its adjacency matrix by $\mathbf{A}$ and random walk matrix by $\mathbf{W}$. Let G_k be the k-subdivision of G, which is defined by replacing every edge of G with a path of length k, effectively adding $k-1$ new vertices of degree 2. While we will focus on analyzing the spectrum of the corresponding matrices $\mathbf{A}_k$ and $\mathbf{W}_k$ of the resulting graph, we clarify already at this point that this process yields a graph of average degree $2+\varepsilon$ by

choosing $k \approx \frac{2(d-2)}{d\varepsilon}$, tying it to our motivation. Nevertheless, the result is more general. The analysis provided here is elementary (i.e., it does not rely on finite free probability), and is inspired by the proofs in [38].

▶ **Lemma 8** (Subdivision and the random walk spectrum). *Let $k \geq 2$, and let G_k be as defined above. Then for every eigenvalue ω of $\mathbf{W}$ and every $t = 0, 1, \ldots, k-1$, $r_t = \cos\left(\frac{\arccos(\omega)+2\pi t}{k}\right)$ is an eigenvalue of $\mathbf{W}_k$. The rest of the eigenvalues are of the form $r_j = \cos\left(\frac{\pi j}{k}\right)$ for $j = 1, \ldots, k-1$.*

The proof uses elementary means, and appears in the full version of this paper.

▶ **Corollary 9.** *Let G be a d-regular graph with normalized one-sided spectral expansion $\omega(G)$, and let S be its subdivision with average degree $2+\varepsilon$. Then*

$$\omega_d(\varepsilon) \;=\; \cos\left(\frac{d\,\varepsilon}{2(d-2)}\,\arccos\left(\omega(G)\right)\right). \tag{8}$$

In particular:

1. *If $d = 3$ and G is Ramanujan, then $\omega_3(\varepsilon) \;=\; \cos\left(\frac{3}{2}\arccos\left(\frac{\sqrt{8}}{3}\right)\varepsilon\right) \;\approx\; 1 - 0.12993\,\varepsilon^2$.*
2. *As $d \to \infty$ (with G Ramanujan), since $\arccos\left(\frac{2\sqrt{d-1}}{d}\right) \to \frac{\pi}{2}$, we obtain $\omega_d(\varepsilon) \longrightarrow \cos\left(\frac{\pi}{4}\,\varepsilon\right) \;=\; 1 - \frac{\pi^2}{32}\,\varepsilon^2 + O(\varepsilon^4)$.*

Proof. Let $k = \left\lceil \frac{(d-2)(2+\varepsilon)}{d\,\varepsilon} \right\rceil$ be the subdivision parameter so that the resulting graph has average degree $2+\varepsilon$. By Lemma 8, the normalized expansion after k-subdivision is $\cos\left(\frac{\arccos(\omega(G))}{k}\right)$. Using the relation between k and ε established in Theorem 3 yields Equation (8).

For (1), put $\omega(G) = \frac{2\sqrt{2}}{3}$ (Ramanujan) and set $\phi \triangleq \arccos(\omega(G)) \approx 0.33984$, giving $\omega_3(\varepsilon) \;=\; \cos\left(\frac{3}{2}\phi\,\varepsilon\right) \;=\; 1 - \frac{(3\phi)^2}{8}\,\varepsilon^2 + O(\varepsilon^3) \;\approx\; 1 - 0.12993\,\varepsilon^2$. For (2), observe that for Ramanujan d-regular graphs, $\omega(G) = \frac{2\sqrt{d-1}}{d}$ and hence $\arccos(\omega(G)) \to \frac{\pi}{2}$ as $d \to \infty$. Plugging into Equation (8) gives the stated limit and its Taylor expansion. ◀

For the unnormalized case, we have the following:

▶ **Lemma 10** (Subdivision and the adjacency matrix spectrum). *Let $k \geq 2$, and let G_k be as defined above. Then every eigenvalue μ of $\mathbf{A}_k$ satisfies one of the conditions below.*

1. *$\mu = 2\cos(\pi j/k)$ for $j = 1, \ldots, k-1$.*
2. *$\mu = 2\cos\theta$ for real θ, satisfying*

$$\lambda = \frac{\sin((k+1)\theta) - (d-1)\,\sin((k-1)\theta)}{\sin\theta}, \tag{9}$$

   where λ is an eigenvalue of $\mathbf{A}$.
3. *$\mu = \pm 2\cosh\xi$ for real $\xi > 0$, satisfying*

$$\lambda = \frac{\sinh((k+1)\xi) - (d-1)\sinh((k-1)\xi)}{\sinh(\xi)}, \tag{10}$$

   where λ is an eigenvalue of $\mathbf{A}$.

Explicit solutions are feasible for $k = 2, 3$, but this is not the case for general k. Nonetheless, We give a sharp estimate of the result for large k, which is the regime of interest in our paper (as $k \approx 1/\varepsilon$ in our desired model).

▶ **Corollary 11.** *All eigenvalues μ of $\mathbf{A}_k$ with $\mu > 2$ satisfy*

$$\mu \;=\; \sqrt{d-1} + \frac{1}{\sqrt{d-1}} \;+\; (1+o_k(1))\,\frac{\lambda}{2}\cdot\frac{(d-2)^2}{\left(\sqrt{d-1}\right)^{k+3}},$$

where λ is an eigenvalue of $\mathbf{A}$.

4 Interlacing Families and Free Convolutions

In this section we review definitions and known results in free probability and interlacing families of polynomials. We use the following standard notation: for a symmetric matrix $\mathbf{A}$, $\chi_x(\mathbf{A})$ is the characteristic polynomial of $\mathbf{A}$ with variable x. We denote by $\lambda_k(\mathbf{A})$ the k-th largest eigenvalue of $\mathbf{A}$ (as $\mathbf{A}$ is symmetric, all eigenvalues are real and hence can be ordered as above). For every symmetric matrix $\mathbf{B}$ for which the all-ones vector $\mathbf{1}$ is an eigenvector and $\mathbf{B1} = b\mathbf{1}$, we denote by $p_{\mathbf{B}}(x)$ the polynomial satisfying $(x-b)p_{\mathbf{B}}(x) = \chi_x(\mathbf{B})$.

For a real-rooted polynomial $p(x)$, we denote by $\alpha_k(p)$ the k-th largest root of $p(x)$. Given a distribution P over polynomials, we denote by $\mathbf{E}_{p\sim P}[p(x)]$ the *expected* polynomial over this distribution, where the expectation is taken in coefficient space, namely, for every k, the coefficient of x^k in $\mathbf{E}_{p\sim P}[p(x)]$ is the expectation over coefficients corresponding to x^k in $p(x)$ drawn according to P.

The following lemma is a simplified version of a more general statement from [23], where it appeared as the main tool enabling us to analyze the expected characteristic polynomial over a distribution, and deduce an existence result.

▶ **Lemma 12.** *Let $\mathbf{A}_1,\ldots,\mathbf{A}_t$ be symmetric $m\times m$ matrices and $(\mathbf{P}_i)_{i\in[t]}$ be uniformly random $m\times m$ permutation matrices. Let $\mathbf{A}_{\mathbf{P}} = \sum_{i=1}^t \mathbf{P}_i\mathbf{A}_i\mathbf{P}_i^{\mathsf{T}}$. Then, for every $k\le m$ there exist permutation matrices $(\mathbf{R}_i)_{i\in[t]}$ such that*

$$\lambda_k(\mathbf{A}_{\mathbf{R}}) \le \alpha_k\left(\mathop{\mathbf{E}}_{\mathbf{P}}\chi_x(\mathbf{A}_{\mathbf{P}})\right). \tag{11}$$

The multiplicative analog of Lemma 12 was proved in [10]. A simplified version, suitable for our needs, is stated below [1].

▶ **Lemma 13.** *Let $\mathbf{M},\mathbf{N}$ be symmetric $m\times m$ matrices such that $\mathbf{M}$ is positive semidefinite, and let $\mathbf{P}$ be a uniformly random $m\times m$ permutation matrix. Then, for every $k\le m$ there exists a permutation matrix $\mathbf{S}$ such that*

$$\lambda_k\left(\mathbf{MSNS}^{\mathsf{T}}\right) \le \alpha_k\left(\mathop{\mathbf{E}}_{\mathbf{P}}\chi_x\left(\mathbf{MPNP}^{\mathsf{T}}\right)\right). \tag{12}$$

▶ **Definition 14** (Additive and multiplicative convolutions)**.** *Let $\mathbf{A},\mathbf{B}$ be real symmetric matrices of dimension m, with characteristic polynomials $a(x) = \chi_x(\mathbf{A})$ and $b(x) = \chi_x(\mathbf{B})$. The* additive convolution *$a\boxplus_m b$ and the* multiplicative convolution *$a\boxtimes_m b$ are the polynomials defined by*

$$(a\boxplus_m b)(x) = \mathop{\mathbf{E}}_{\mathbf{Q}}\chi_x\left(\mathbf{A}+\mathbf{QBQ}^{\mathsf{T}}\right),$$

[1] Lemma 13 was proved in [10, Theorem 6.5 and Lemma 6.3] in more generality (for a sequence of permutation matrices, as in Lemma 12). It is phrased therein, however, only for the case $\mathbf{M} = \mathbf{A}^2$ and $\mathbf{N} = \mathbf{A}$ for a symmetric matrix $\mathbf{A}$.

and

$$(a \boxtimes_m b)(x) = \mathop{\mathbf{E}}_{\mathbf{Q}} \chi_x \left(\mathbf{A}\mathbf{Q}\mathbf{B}\mathbf{Q}^{\mathsf{T}}\right),$$

where $\mathbf{Q}$ *is a Haar random orthogonal matrix. We will use* $\boxplus$ *and* $\boxtimes$ *instead of* $\boxplus_m$ *and* $\boxtimes_m$ *for simplifying notation when the dimension is clear from the context.*

Working with graph matrices, an issue with the above definition is that the Haar measure does not have a meaningful combinatorial interpretation. Therefore, we need a way to relate *permutation* matrices - which do have such interpretation - to Haar random matrices. To this end we state the following lemma (which appeared first in [23] for the additive case, and later on in [10] for the multiplicative case, whose proof uses similar ideas) which is described as a *Quadrature* result, translating an infinite (continuous) measure space to a finite one.

▶ **Lemma 15** (*Quadrature*; Corollary 4.9 from [23] and Lemma 2.3 from [10])**.** *Let* $\mathbf{A}, \mathbf{B}$ *be real* $m \times m$ *symmetric matrices such that* $\mathbf{A}\mathbf{1} = a\mathbf{1}$ *and* $\mathbf{B}\mathbf{1} = b\mathbf{1}$*. Denote by* $p_{\mathbf{A}}, p_{\mathbf{B}}$ *the polynomials satisfying* $\chi_x(\mathbf{A}) = (x-a)p_{\mathbf{A}}(x)$*,* $\chi_x(\mathbf{B}) = (x-b)p_{\mathbf{B}}(x)$*. Let* $\mathbf{P}$ *be a uniformly random* $m \times m$ *permutation matrix. Then,*

$$\mathop{\mathbf{E}}_{\mathbf{P}} \chi_x \left(\mathbf{A} + \mathbf{P}\mathbf{B}\mathbf{P}^{\mathsf{T}}\right) = (x - (a+b)) \left(p_{\mathbf{A}} \boxplus p_{\mathbf{B}}\right)(x), \tag{13}$$

$$\mathop{\mathbf{E}}_{\mathbf{P}} \chi_x \left(\mathbf{A}\mathbf{P}\mathbf{B}\mathbf{P}^{\mathsf{T}}\right) = (x - ab) \left(p_{\mathbf{A}} \boxtimes p_{\mathbf{B}}\right)(x). \tag{14}$$

4.1 Transforms

Let μ be a compactly supported probability distribution over $\mathbb{R}$, and let $a = \sup(\mu)$. The *Cauchy transform* of μ is defined on the domain (a, ∞) as the function

$$\mathcal{G}_\mu(x) = \int_{\mathbb{R}} \frac{1}{x-t} \mu(t) dt.$$

We remark that in many settings it is instructive to study the Cauchy transform as a function whose domain is $\mathbb{C}^+$. However, we will consider the Cauchy transform as a function on $\mathbb{R}$. The Cauchy transform is also related to the moments of a distribution. If $m_r(\mu)$ is the r-th moment of μ, then for every $x > a$ we have (see [27, Remark 2.19]):

$$\mathcal{G}_\mu(x) = \sum_{r=0}^{\infty} \frac{m_r(\mu)}{x^{r+1}}. \tag{15}$$

Let $p(x)$ be a degree m real-rooted polynomial with roots $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_m$. To $p(x)$ we associate the uniform distribution over its roots, and therefore its Cauchy transform (defined for $x > \alpha_1$) is

$$\mathcal{G}_p(x) = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{x - \alpha_i} = \frac{1}{m} \cdot \frac{p'(x)}{p(x)}.$$

Accompanying the Cauchy transform of μ is the *$\mathcal{M}$-transform* (or moment transform), which is defined by $\mathcal{M}_\mu(x) = x\mathcal{G}_\mu(x) - 1$. It is easy to see that $\mathcal{G}_\mu(x)$ is monotonically decreasing within its domain, and thus invertible. Denote the range of $\mathcal{G}_\mu(x)$ by $(0, L^+)$, that is, $L^+ = \lim_{x \to +a} \mathcal{G}_\mu(x)$ (for a polynomial $p(x)$, $L^+ = \infty$). With this in mind, we

define $\mathcal{K}_\mu : (0, L^+) \to (a, \infty)$ as the inverse of $\mathcal{G}_\mu$. In cases where $\mathcal{M}_\mu(x)$ is monotonically decreasing in the same domain, we can define $\mathcal{N}_\mu(y)$ as the inverse of $\mathcal{M}_\mu$ in a similar manner. By definition, for every $y \in (0, L^+)$, both $\mathcal{K}_\mu(y)$ and $\mathcal{N}_\mu(y)$ provide an upper bound on a, the supremum of the support of μ. In case of a polynomial $p(x)$, this bounds the largest root of $p(x)$, which we state for later reference in the following claim:

▷ Claim 16. Let $p(x)$ be a degree m real-rooted polynomial with roots $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_m$. Then for every $y \in (0, \infty)$, both $\alpha_1 \leq \mathcal{K}_p(y)$ and $\alpha_1 \leq \mathcal{N}_p(y)$.

For a symmetric matrix $\mathbf{A}$, we use the notation $\mathcal{G}_\mathbf{A}$ for $\mathcal{G}_{\chi_x(\mathbf{A})}$, and $\mathcal{G}_{p_\mathbf{A}}$ for the Cauchy transform of $p_\mathbf{A}$ as used in Lemma 15 if applicable, and similar notations for the other transforms.

The key feature of the $\mathcal{K}$ and the $\mathcal{N}$ transforms is that they behave very well under additive and multiplicative convolutions, respectively, as was shown in [24, Theorem 1.11] and [9, Claim 4.12], and stated below.

▶ **Lemma 17** (Convolution bounds). *For real-rooted polynomials $p(x)$ and $q(x)$ of degree m, and for any $y > 0$,*

$$\mathcal{K}_{p \boxplus_m q}(y) \leq \mathcal{K}_p(y) + \mathcal{K}_q(y) - \frac{1}{y}.$$

When one of the polynomials has only non-negative roots, then for every $y > 0$,

$$\mathcal{N}_{p \boxtimes_m q}(y) \leq \frac{y}{y+1} \cdot \mathcal{N}_p(y) \cdot \mathcal{N}_q(y).$$

Both inequalities are strict in case both polynomials have at least two distinct roots.

An immediate corollary of Lemma 17 and Claim 16 is the following.

▶ **Corollary 18.** *For every real-rooted polynomial $p(x)$, $t \in \mathbb{N}$ and $y > 0$,* $\mathsf{maxroot}\left(p^{\boxplus t}(x)\right) \leq t \cdot \mathcal{K}_p(y) - \frac{t-1}{y}$.

▷ Claim 19. [9, Claim 4.12] Let $\mathbf{B}$ be a symmetric matrix such that $\mathbf{B}\mathbf{1} = b\mathbf{1}$ and b is its largest eigenvalue. Then, for every $x > b$, $\mathcal{G}_{p_\mathbf{B}}(x) \leq \mathcal{G}_\mathbf{B}(x)$ and $\mathcal{M}_{p_\mathbf{B}}(x) \leq \mathcal{M}_\mathbf{B}(x)$, and for every $y > 0$, $\mathcal{K}_{p_\mathbf{B}}(y) \leq \mathcal{K}_\mathbf{B}(y)$ and $\mathcal{N}_{p_\mathbf{B}}(y) \leq \mathcal{N}_\mathbf{B}(y)$.
When $\mathbf{B} \neq b\mathbf{I}$, all inequalities are sharp.

5 Ultra-Sparse Expanders in the Configuration Model

In this section we prove Theorem 1, which immediately implies Corollary 2.

▶ **Definition 20** (Configuration model for regular graphs). *A random undirected d-regular graph on n vertices is said to be sampled by the* configuration model *if it is generated by choosing a uniformly random perfect matching on the nd* half-edges.

5.1 Alternate MSS proof via finite free projection

In this section, we present our alternative proof of the MSS result using the configuration model.

▶ **Theorem 21.** *For every $d \geq 2, n \in \mathbb{N}$ such that $N = nd$ is even, there exists an $N \times N$ permutation matrix $\mathbf{P}$ such that the d-regular n-vertex graph $G_\mathbf{P}$ satisfies*

$$\lambda_2(G_\mathbf{P}) < 2\sqrt{d-1}.$$

Proof. We begin with the useful observation that

$$\mathbf{V} \triangleq \mathbf{U}\mathbf{U}^\mathsf{T} = \mathbf{J}_d \otimes \mathbf{I}_n,$$

where $\mathbf{J}_d$ denotes the $d \times d$ matrix with all entries equal to 1. Note that $\mathbf{J}_d$ is the random walk matrix (i.e., the normalized adjacency matrix) of the complete graph on d vertices with self-loops. Observe that

$$\chi_x(\mathbf{M}) = (x-1)^{\frac{N}{2}}(x+1)^{\frac{N}{2}},$$
$$\chi_x(\mathbf{V}) = x^{n(d-1)}(x-d)^n,$$

yielding the $\mathcal{M}$-transforms

$$\mathcal{M}_{\mathbf{M}}(x) = \frac{1}{x^2-1}, \quad \mathcal{M}_{\mathbf{V}}(x) = \frac{1}{x-d}. \tag{16}$$

As both are decreasing in their respective domains $(1,\infty)$, (d,∞), we can calculate their $\mathcal{N}$-transforms (see Section 4.1) and get

$$\mathcal{N}_{\mathbf{M}}(y) = \sqrt{\frac{y+1}{y}}, \quad \mathcal{N}_{\mathbf{V}}(y) = \frac{dy+1}{y}. \tag{17}$$

Now, for any matrix $\mathbf{P}$,

$$x^{n(d-1)}\chi_x\left(\mathbf{U}^\mathsf{T}\mathbf{P}\mathbf{M}\mathbf{P}^\mathsf{T}\mathbf{U}\right) = \chi_x\left(\mathbf{U}\mathbf{U}^\mathsf{T}\mathbf{P}\mathbf{M}\mathbf{P}^\mathsf{T}\right),$$

since characteristic polynomials are invariant under cyclic rotations – up to a multiplicative factor of x, depending on the difference in dimensions. By Lemma 13, it is enough to prove that

$$\alpha_2 \triangleq \alpha_2\left(\mathop{\mathbf{E}}_{\mathbf{P}} \chi_x\left(\mathbf{U}\mathbf{U}^\mathsf{T}\mathbf{P}\mathbf{M}\mathbf{P}^\mathsf{T}\right)\right) < 2\sqrt{d-1}, \tag{18}$$

where recall that for a real-rooted polynomial $p(x)$, we let $\alpha_i(p(x))$ denote its i-th largest root. By Lemma 15,

$$\mathop{\mathbf{E}}_{\mathbf{P}} \chi_x\left(\mathbf{U}\mathbf{U}^\mathsf{T}\mathbf{P}\mathbf{M}\mathbf{P}^\mathsf{T}\right) = (x-d)\left(p_{\mathbf{V}} \boxtimes p_{\mathbf{M}}\right)(x), \tag{19}$$

where $p_{\mathbf{V}}(x)$ and $p_{\mathbf{M}}(x)$ are the polynomials defined by $\chi_x(\mathbf{V}) = (x-d)p_{\mathbf{V}}(x)$ and $\chi_x(\mathbf{M}) = (x-1)p_{\mathbf{M}}(x)$, respectively. Therefore,

$$\mathcal{N}_{p_{\mathbf{V}} \boxtimes p_{\mathbf{M}}}(y) \le \frac{y}{y+1}\mathcal{N}_{p_{\mathbf{M}}}(y)\mathcal{N}_{p_{\mathbf{V}}}(y) < \frac{dy+1}{\sqrt{(y(y+1))}}, \tag{20}$$

where the first inequality follows by Lemma 17, and the second, sharp, inequality follows by Claim 19 and the calculations of $\mathcal{N}_{\mathbf{M}}(y), \mathcal{N}_{\mathbf{V}}(y)$ above.

By Claim 16, $\alpha_2 \le \mathcal{N}_{p_{\mathbf{V}} \boxtimes p_{\mathbf{M}}}(y)$ for any $y > 0$. Choosing the point $y_0 = \frac{1}{d-2}$, we get by the above inequality that

$$\alpha_2 \le \mathcal{N}_{p_{\mathbf{V}} \boxtimes p_{\mathbf{M}}}(y_0) < \frac{dy_0+1}{\sqrt{y_0(y_0+1)}} = 2\sqrt{d-1}. \qquad \blacktriangleleft$$

5.2 Non-regular graphs in the configuration model

In this section, we extend our alternative proof of MSS's theorem, presented in the previous section, to the setting of irregular graphs. The configuration model is well-suited for handling vertices with varying degrees, and the required changes are minor, mostly involving appropriate normalizations.

Before proceeding, we recall the definition of the random walk matrix for irregular graphs. First, recall that the random walk matrix of a d-regular graph G, with adjacency matrix $\mathbf{A}$, is given by $\mathbf{W} = \frac{1}{d}\mathbf{A}$, and we denote its eigenvalues by $1 = \omega_1 \geq \omega_2 \geq \cdots \geq \omega_n \geq -1$. The *normalized spectral expansion* of G, denoted $\omega(G)$, is defined as $\max(|\omega_2|, |\omega_n|) \in [0,1]$. When G is *not* regular, the random walk matrix is defined as $\mathbf{W} = \mathbf{A}\mathbf{D}^{-1}$, where $\mathbf{D}$ is the diagonal matrix of vertex degrees (we assume no isolated vertices). This generalizes the above definition for regular graphs. The parameter $\omega(G)$ governs the convergence rate of random walks on G. It is easy to see that the eigenvalues of $\mathbf{W}$ are the same as those of the symmetric matrix $\mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$. Since we are primarily interested in the eigenvalues, we will treat this symmetric matrix as the random walk matrix in what follows.

To derive the analogue in the irregular case, we introduce the following notation. Let $d_1, \ldots, d_n$ be a degree sequence such that $N = \sum_{i=1}^{n} d_i$ is even. Define $j_k = \sum_{i=1}^{k} d_i$ as the sum of the first k degrees, and let $J_k \subset [N]$ be the set of indices

$$J_k = \{j_{k-1} + 1, j_{k-1} + 2, \ldots, j_{k-1} + d_k\}.$$

The analogue of the matrix $\mathbf{U}$ in the non-regular case is the $N \times n$ matrix $\mathbf{C}$ defined by $\mathbf{C}e_k = \frac{1}{\sqrt{d_k}}\mathbf{1}_{J_k}$ for every $k \in [n]$. Using this notation, we obtain

$$\mathbf{W}_{\mathbf{P}} = \mathbf{C}^{\mathsf{T}}\mathbf{P}\mathbf{M}\mathbf{P}^{\mathsf{T}}\mathbf{C}, \tag{21}$$

where $\mathbf{M}$ is the adjacency matrix of a perfect matching on the N half-edges, as in the regular configuration model. With this, we are ready to prove Theorem 1. The proof follows similar lines to that of Theorem 21, and so we refrain from repeating it in full, instead focusing on the differences.

Proof of Theorem 1. First, observe that $\mathbf{C}\mathbf{C}^{\mathsf{T}}$ is the block-diagonal matrix, which we denote by $\bar{\mathbf{V}}$, consisting of n blocks of the form $\frac{1}{d_i}\mathbf{J}_{d_i}$. Note that

$$\chi_x\left(\bar{\mathbf{V}}\right) = (x-1)^n x^{N-n}.$$

A straightforward calculation gives $\mathcal{N}_{\bar{\mathbf{V}}}(y) = \frac{\bar{d}y+1}{\bar{d}y}$. Following the same steps as in the proof of Theorem 21, we arrive at the analogue of Equation (20),

$$\mathcal{N}_{p_{\bar{\mathbf{V}}} \boxtimes p_{\mathbf{M}}}(y) \leq \frac{y}{y+1} \cdot \mathcal{N}_{p_{\mathbf{M}}}(y) \cdot \mathcal{N}_{p_{\bar{\mathbf{V}}}}(y) < \frac{\bar{d}y+1}{\bar{d}\sqrt{y(y+1)}}.$$

The proof follows as the right-hand side attains its minimum value $\frac{2\sqrt{\bar{d}-1}}{\bar{d}}$ (at $y_0 = \frac{1}{\bar{d}-2}$). ◀

6 The Free Method

The analysis in Section 5.2 deals with non-regular graphs directly for the normalized spectrum (of the matrix $\mathbf{W}$), which governs the convergence of random walks (among other characteristics). This raises the question of what can be said about the adjacency spectrum of $\mathbf{A}$, which is of importance when analyzing various generalizations of the expander mixing lemma (see [1] for an up to date survey of the latter).

There are several frameworks for using results from free probability to deduce analysis on graph distributions. One of these is the *finite free* approach using interlacing families of polynomials, pioneered in [23] (and later used in [26, 10] and others), and exemplified here in Section 5.2. Others are the strong convergence approaches of Bordenave and Collins [7] and Chen et al. [8]. All these approaches can be looked at as a two-step process:

1. A reduction from a combinatorial question to a setting of free random variables.
2. Solution in the free setting.

The reduction of Item 1 can yield either an existence proof (as in the MSS approach used here in Theorem 1), or a high probability result as in [7, 8], reduced to the solution of Item 2. The latter, however, is an elusive topic which did not get nearly enough attention in the literature, and we overview it next. We then analyze specific examples in the free regime and verify that the results well represent the behavior of random graphs.

6.1 Free method procedures

In this section, we present a couple of procedures within the framework of the free method. We begin with a brief overview of techniques used in free probability for computing distributions of sums and products of free random variables. In our context, we think of these as the eigenvalue distributions of matrices, for which the eigenspaces are maximally decoupled (for example, by rotating one matrix using a Haar unitary). However, in the free probability literature this context is not necessary, as there are clean self-contained definitions of free random variables expressed in terms of joint moments (for an excellent textbook on the topic, we refer the reader to [27]).

Let a and b be *free* random variables, and denote their distributions by μ_a and μ_b. We wish to find the distribution μ_c of the variable $c = a + b$, by which we define the free additive convolution of distributions $\mu_c = \mu_a \boxplus \mu_b$. This represents, in matrix terms, the eigenvalue distribution of $\mathbf{A} + \mathbf{Q}^\mathsf{T}\mathbf{B}\mathbf{Q}$ where $\mathbf{Q}$ is a Haar unitary and $\mathbf{A}, \mathbf{B}$ are square matrices with eigenvalue distributions μ_a, μ_b respectively. The recipe doing so is the following, provided here without proof[2]:

Algorithm 1. Free additive convolution.

1. Calculate the Cauchy transforms $\mathcal{G}_{\mu_a}(x)$ and $\mathcal{G}_{\mu_b}(x)$ as defined in Section 4.1.
2. Find the compositional inverses, denoted by $\mathcal{K}_{\mu_a}(y)$ and $\mathcal{K}_{\mu_b}(y)$ (as defined in Section 4.1).
3. Calculate $\mathcal{K}_{\mu_c}(y) = \mathcal{K}_{\mu_a}(y) + \mathcal{K}_{\mu_b}(y) - \frac{1}{y}$.
4. Invert $\mathcal{K}_{\mu_c}(y)$ to get $\mathcal{G}_{\mu_c}(x)$.
5. Calculate μ_c from $\mathcal{G}_{\mu_c}(x)$ using the Stieltjes inversion formula (details omitted).

Note that when at least one of the distributions is supported on non-negative reals only (in the context of matrices, at least one of them is PSD) there is a multiplicative analog of the above recipe – that is, the calculation of the distribution $\mu_c = \mu_a \boxtimes \mu_b$ for $c = ab$ – where the $\mathcal{G}$ and $\mathcal{K}$ transforms are replaced with $\mathcal{M}$, $\mathcal{N}$ respectively, and Item 3 is replaced with $\mathcal{N}_{\mu_c}(y) \leq \frac{y}{y+1} \cdot \mathcal{N}_{\mu_a}(y) \cdot \mathcal{N}_{\mu_b}(y)$.

The vast majority of the literature focuses on proofs of the above, and its generalizations. However, explicit applications of it as stated are not very common, due to a few clear caveats. The first is that for Item 1 we need to have explicit analytic formulas for μ_a, μ_b, which

[2] The proof is standard in the free probability literature, and can be found in [27, Chapter 12].

may not be available. The second is that inverting a function is a complicated, sometimes infeasible, task. Given a distribution μ supported on only 3 values, inverting $\mathcal{G}_\mu$ involves solving a cubic equation, which may be possible but leads to a function that is hard to analyze. For more complicated distributions, we know the task to be infeasible, making Items 2 and 4 mere theoretical ideas and not practical tools.

A solution to the inversion problem of Item 4 is implicit in the work of Marcus, Spielman and Srivastava. Once we relax our goal of having an explicit formula for μ_c and wish to only *bound* its support, the only necessary part is knowing $\mathcal{K}_{\mu_c}(y)$. This is due to the observation (discussed here in Section 4.1) that every value of this function is an upper bound on the support. Therefore Items 4 and 5 can be replaced with minimizing $\mathcal{K}_{\mu_c}(y)$ over $y > 0$, which is typically a simple analytic task, leading to the following simpler procedure:

Algorithm 2. Bounding the free additive convolution.

- Perform Items 1–3 of Procedure 1 to get $\mathcal{K}_{\mu_c}(y)$.
- Output $\min_{y>0} \mathcal{K}_{\mu_c}(y)$.

Although this procedure is simpler, we are still left with the case of Item 2 not being feasible, which is the issue we wish to solve next in Section 6.1.1. Note that other works have looked into relaxing the requirement to get a closed formula for μ_c, to rather learn characteristics of its support [3] or develop numerical techniques for this task (see [12, 29] and the subordination technique presented in [34]). To the best of our knowledge, although the technique presented here is elementary, this work is the first to formally define it.

6.1.1 Sums of identical spectra

The procedure given in the next lemma solves the inversion issue of Item 2 for the case that the random variables have an equal distribution μ. We use $\mu^{\boxplus t}$ to denote the additive convolution of μ with itself $t-1$ times.

▶ **Lemma 22.** *Let μ be a compactly supported distribution on $\mathbb{R}$, and $t \in \mathbb{N}$. Then the supremum of the support of $\mu^{\boxplus t}$ is achieved by minimizing*

$$\widetilde{\mathcal{K}_t}(x) \triangleq tx - \frac{t-1}{\mathcal{G}_\mu(x)}$$

over $\{x > \max(\mu)\}$, or equivalently applying the following procedure:

Algorithm 3. Bounding the support of $\mu^{\boxplus t}$.

1. Find the solution x_0 of the equation
$$\frac{t-1}{t}\mathcal{G}'_\mu(x) = -\mathcal{G}_\mu(x)^2. \tag{22}$$
2. Evaluate $y_0 \triangleq \mathcal{G}_\mu(x_0)$.
3. Output $t \cdot x_0 - \frac{t-1}{y_0}$.

Notice that at no point in Lemma 22 do we need to invert a function.

Proof. By definition of inverse under composition, $\mathcal{G}_\mu(\mathcal{K}_\mu(y)) = y$. Differentiating both sides and rearranging gives

$$\mathcal{K}'_\mu(y) = \frac{1}{\mathcal{G}'_\mu(\mathcal{K}_\mu(y))}. \tag{23}$$

As shown, we look for the minimal value of

$$\mathcal{K}_{\mu_t}(y) = t \cdot \mathcal{K}_\mu(y) - \frac{t-1}{y} \tag{24}$$

achieved at a point y_0 satisfying $\mathcal{K}'_{\mu_t}(y_0) = 0$. Note that such a minimum always exists, as $\mathcal{G}_\mu(x)$ decays like $\frac{1}{x}$. Therefore by Equation (24)

$$t \cdot \mathcal{K}'_\mu(y_0) = -\frac{t-1}{y_0^2}.$$

Plugging in Equation (23) one concludes

$$\frac{t}{\mathcal{G}'_\mu(\mathcal{K}_\mu(y_0))} = -\frac{t-1}{y_0^2}.$$

Let x_0 be the value such that $y_0 = \mathcal{G}_\mu(x_0)$. Then we get from the above that

$$\frac{t-1}{t}\mathcal{G}'_\mu(x_0) = -\mathcal{G}(x_0)^2,$$

as required in Equation (22). The minimal value of Equation (24) is thus

$$\mathcal{K}_{\mu_t}(y_0) = t \cdot \mathcal{K}_\mu(y_0) - \frac{t-1}{y_0} = t \cdot x_0 - \frac{t-1}{y_0}.$$ ◀

6.1.2 Sums and products of two distinct spectra

The procedure of Lemma 22 is very easy to use, but does not cover cases where we need to perform an additive convolution of different distributions μ_1 and μ_2. A straightforward variation of Lemma 22 shows that when one distribution has a simple Cauchy transform $\mathcal{G}_{\mu_1}(x)$ with an explicit inverse $\mathcal{K}_{\mu_1}(y)$, the supremum of the support of $\mu_1 \boxplus \mu_2$ can be found using the following lemma.

▶ **Lemma 23.** *Let μ_1, μ_2 be compactly supported distributions on $\mathbb{R}$. The supremum of the support of $\mu_1 \boxplus \mu_2$ is achieved by minimizing over $\{x > \max(\mu_2)\}$ the function*

$$\widetilde{\mathcal{K}}_{\mu_1 \boxplus \mu_2}(x) \triangleq \mathcal{K}_{\mu_1}(\mathcal{G}_{\mu_2}(x)) + x - \frac{1}{\mathcal{G}_{\mu_2}(x)}. \tag{25}$$

If either μ_1 or μ_2 is supported on $\mathbb{R}^+$ only, the supremum of the support of $\mu_1 \boxtimes \mu_2$ is achieved by minimizing over $\{x > \max(\mu_2)\}$ the function

$$\widetilde{\mathcal{N}}_{\mu_1 \boxtimes \mu_2}(x) \triangleq \frac{\mathcal{M}_{\mu_2}(x)}{\mathcal{M}_{\mu_2}(x)+1} \cdot \mathcal{N}_{\mu_1}(\mathcal{M}_{\mu_2}(x)) \cdot x. \tag{26}$$

Proof. We will prove the additive case, the multiplicative follows similar arguments. By Item 3 of Procedure 1, we need to minimize

$$\mathcal{K}_{\mu_1 \boxplus \mu_2}(y) = \mathcal{K}_{\mu_1}(y) + \mathcal{K}_{\mu_2}(y) - \frac{1}{y}.$$

Set $y = \mathcal{G}_{\mu_2}(x)$, and we get the equation

$$\widetilde{\mathcal{K}}_{\mu_1 \boxplus \mu_2}(x) = \mathcal{K}_{\mu_1}(\mathcal{G}_{\mu_2}(x)) + x - \frac{1}{\mathcal{G}_{\mu_2}(x)},$$

completing the proof. ◀

6.2 Application to ultra-sparse graphs

As previewed in Section 2.2, we predict spectral bounds for our graph models by analyzing the spectra of their constituent pieces and how they combine. Our prediction for the resulting graph is the free convolution of the component spectral measures – additive ($\boxplus$) or multiplicative ($\boxtimes$), depending on the model. For each model we apply the appropriate theorem from Section 6.1 and express the bound as a function of the sparsity ε, denoted $\mathsf{pred}_{\mathsf{model}}(\varepsilon)$, where we substitute the suitable model in the subscript.

Free method predictions for the configuration model

As mentioned above, in order to apply the techniques of Section 6.1 to random graph models, one needs to understand the distributions involved. For the configuration model, the convolution is of the eigenvalue distribution of the matrix $\mathbf{M}$, which we already know to be the uniform distribution on ± 1 and we denote by μ_1, and the one of $\mathbf{V} = \mathbf{U}\mathbf{U}^{\mathsf{T}}$ which we denote by μ_2. The latter's form is defined by the distribution of degrees. In the case of a graph with degrees 2 and 3 and average degree $2+\varepsilon$, a straightforward calculation shows its Cauchy transform to be

$$\mathcal{G}_{\mu_2}(x) = \frac{1}{2+\varepsilon}\left(\frac{1+\varepsilon}{x} + \frac{\varepsilon}{x-3} + \frac{1-\varepsilon}{x-2}\right).$$

By Equation (17) we have that $\mathcal{N}_{\mu_1}(y) = \sqrt{\frac{y+1}{y}}$. Thus, applying Lemma 23 gives us that $\sup(\mu_1 \boxtimes \mu_2)$ is found by minimizing over $x > 3$ the function

$$\widetilde{\mathcal{N}}_{\mu_1 \boxtimes \mu_2}(x) = x \cdot \sqrt{\frac{\mathcal{M}_{\mu_2}(x)}{\mathcal{M}_{\mu_2}(x)+1}} = x \cdot \sqrt{\frac{x\mathcal{G}_{\mu_2}(x)-1}{x\mathcal{G}_{\mu_2}(x)}}. \tag{27}$$

This leads us to the main result of this section.

▶ **Lemma 24** (Small–ε bound for $\mathsf{pred}_{\mathsf{conf}}(\varepsilon)$)**.**

$$\mathsf{pred}_{\mathsf{conf}}(\varepsilon) = \frac{3}{\sqrt{2}} + \frac{3}{4}\varepsilon^{1/2} + \frac{9\sqrt{2}}{32}\varepsilon - O(\varepsilon^{3/2}).$$

The proof uses elementary means and appears in the full version of this paper.

Free method predictions for the $\mathcal{C} + \varepsilon\mathcal{M}$, $(2+\varepsilon)\mathcal{M}$ and $3\left(\frac{2+\varepsilon}{3}\mathcal{M}\right)$ models appear in the full version of the paper.

References

1 Aida Abiad and Sjanne Zeijlemaker. A unified framework for the expander mixing lemma for irregular graphs and its applications. *Linear Algebra and its Applications*, 702:19–45, 2024.

2 Noga Alon, Jeff Edmonds, and Michael Luby. Linear time erasure codes with nearly optimal recovery. In *36th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 512–519, 1995.

3 Zhigang Bao, László Erdős, and Kevin Schnelli. On the support of the free additive convolution. *Journal d'Analyse Mathématique*, 142(1):323–348, 2020.

4 Avraham Ben-Aroya and Amnon Ta-Shma. A combinatorial construction of almost-Ramanujan graphs using the zig-zag product. *SIAM J. Comput.*, 40(2):267–290, 2011. doi:10.1137/080732651.

5 Yonatan Bilu and Nathan Linial. Lifts, discrepancy and nearly optimal spectral gap. *Combinatorica*, 26(5):495–519, 2006. doi:10.1007/s00493-006-0029-7.

6 Charles Bordenave. A new proof of Friedman's second eigenvalue theorem and its extension to random lifts. *Ann. Sci. Éc. Norm. Supér. (4)*, 53(6):1393–1439, 2020. doi:10.24033/asens.2450.

7 Charles Bordenave and Benoît Collins. Eigenvalues of random lifts and polynomials of random permutation matrices. *Annals of Mathematics*, 190(3):811–875, 2019.

8 Chi-Fang Chen, Jorge Garza-Vargas, Joel A Tropp, and Ramon van Handel. A new approach to strong convergence. *Annals of Mathematics*, 2024.

9 Gil Cohen, Itay Cohen, and Gal Maor. Tight bounds for the zig-zag product. In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, 2024.

10 Gil Cohen and Gal Maor. Random walks on rotating expanders. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, 2023.

11 Michael B. Cohen. Ramanujan graphs in polynomial time. In *57th Annual IEEE Symposium on Foundations of Computer Science – FOCS 2016*, pages 276–281. IEEE Computer Soc., Los Alamitos, CA, 2016. doi:10.1109/FOCS.2016.37.

12 Alice Cortinovis and Lexing Ying. Computing free convolutions via contour integrals. *arXiv preprint*, 2023. doi:10.48550/arXiv.2305.01819.

13 Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):Art. 12, 44, 2007. doi:10.1145/1236457.1236459.

14 Irit Dinur and Oded Goldreich. Expansion without connectivity: A property testing perspective. *ECCC*, 2025.

15 Shlomo Hoory. A lower bound on the spectral radius of the universal cover of a graph. *Journal of Combinatorial Theory, Series B*, 93(1):33–43, 2005. doi:10.1016/J.JCTB.2004.06.001.

16 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.

17 Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 356–364, 1994. doi:10.1145/195058.195190.

18 Kazimierz Kuratowski. Sur le problème des courbes gauches en topologie. *Fundamenta Mathematicae*, 15:271–283, 1930. doi:10.4064/fm-15-1-271-283.

19 Alexander Lubotzky. Finite simple groups of Lie type as expanders. *Journal of the European Mathematical Society*, 13(5):1331–1341, 2011.

20 Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988. doi:10.1007/BF02126799.

21 Michael G Luby, Michael Mitzenmacher, Mohammad Amin Shokrollahi, and Daniel A Spielman. Improved low-density parity-check codes using irregular graphs. *IEEE Transactions on information Theory*, 47(2), 2002. doi:10.1109/18.910576.

22 Adam W. Marcus, Daniel A. Spielman, and Nikhil Srivastava. Interlacing families I: Bipartite Ramanujan graphs of all degrees. *Ann. of Math. (2)*, 182(1):307–325, 2015. doi:10.4007/annals.2015.182.1.7.

23 Adam W Marcus, Daniel A Spielman, and Nikhil Srivastava. Interlacing families IV: Bipartite ramanujan graphs of all sizes. *SIAM Journal on Computing*, 47(6):2488–2509, 2018. doi:10.1137/16M106176X.

24 Adam W. Marcus, Daniel A. Spielman, and Nikhil Srivastava. Finite free convolutions of polynomials. *Probab. Theory Related Fields*, 182(3-4):807–848, 2022. doi:10.1007/s00440-021-01105-w.

25 G. A. Margulis. Explicit constructions of graphs without short cycles and low density codes. *Combinatorica*, 2(1):71–78, 1982. doi:10.1007/BF02579283.

26 Sidhanth Mohanty and Ryan O'Donnell. X-Ramanujan graphs. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms*, pages 1226–1243. SIAM, Philadelphia, PA, 2020. doi:10.1137/1.9781611975994.75.

27 Alexandru Nica and Roland Speicher. *Lectures on the combinatorics of free probability*, volume 335 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, Cambridge, 2006. doi:10.1017/CBO9780511735127.

28 A. Nilli. On the second eigenvalue of a graph. *Discrete Math.*, 91(2):207–210, 1991. doi:10.1016/0012-365X(91)90112-F.

29 Sheehan Olver and Raj Rao Nadakuditi. Numerical computation of convolutions in free probability theory. *arXiv preprint*, 2012. arXiv:1203.1958.

30 Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4):Art. 17, 24, 2008. doi:10.1145/1391289.1391291.

31 Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 3–13. IEEE, 2000. doi:10.1109/SFCS.2000.892006.

32 Thomas J Richardson, Mohammad Amin Shokrollahi, and Rüdiger L Urbanke. Design of capacity-approaching irregular low-density parity-check codes. *IEEE transactions on information theory*, 47(2), 2002. doi:10.1109/18.910578.

33 Michael Sipser and Daniel A. Spielman. Expander codes. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 566–576, 1994. doi:10.1109/SFCS.1994.365734.

34 Roland Speicher. Lecture notes on free probability theory. *arXiv preprint*, 2019. arXiv:1908.08125.

35 Daniel A. Spielman. Spectral and algebraic graph theory. Unpublished book manuscript, 2025. URL: http://cs-www.cs.yale.edu/homes/spielman/sagt/sagt.pdf.

36 Amnon Ta-Shma. Explicit, almost optimal, epsilon-balanced codes. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 238–251, 2017. doi:10.1145/3055399.3055408.

37 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/0400000010.

38 Pinchen Xie, Zhongzhi Zhang, and Francesc Comellas. The normalized laplacian spectrum of subdivisions of a graph. *Applied Mathematics and Computation*, 286:250–256, 2016. doi:10.1016/J.AMC.2016.04.033.

Entropy Equivalence Testing

Clément L. Canonne
University of Sydney, Australia

Yash Pote
National University of Singapore, Singapore

Jonathan Scarlett
National University of Singapore, Singapore

Joy Qiping Yang
University of Sydney, Australia

Abstract

We introduce the problem of *entropy equivalence testing* for probability distributions, a relaxation of the well-studied closeness testing problem, where the distribution testing algorithm is now only required to distinguish, given samples from two unknown distributions p, q and a parameter $\varepsilon \in (0, 1/2]$, between $p = q$ and $|H(p) - H(q)| \geqslant \varepsilon$ (where H denotes the Shannon entropy). We provide a time- and sample-efficient algorithm for this task, showing that the optimal sample complexity for this task can be significantly lower than that of closeness testing. As an application, we leverage this result to provide the first non-trivial testing algorithm for (standard) closeness of low-degree *Bayesian networks*, which significantly improves on either the sample or time complexity of a baseline based on full learning.

2012 ACM Subject Classification Theory of computation → Lower bounds and information complexity; Mathematics of computing → Probabilistic algorithms

Keywords and phrases Entropy, distribution testing, sublinear algorithm, Bayesian network

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.50

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2605.23225` [14]

Funding *Jonathan Scarlett*: Supported by the National University of Singapore under the Presidential Young Professorship scheme.
Joy Qiping Yang: Supported by a JD Technology Research Scholarship in Artificial intelligence.

Acknowledgements We would like to thank Sayantan Sen, Jan Seyfried and Marco Tomamichel for the discussion on their work of testing Conditional Mutual Information. JY would also like to thank Davin Choo for the early discussions predating this project. This work is conducted while JY was interning at National Institute of Informatics, hosted by Yuichi Yoshida.

1 Introduction

Distribution testing, a subfield of property testing [35, 30] concerned with efficient algorithms for deciding statistical properties of data, has witnessed a significant interest since its systematic introduction by Batu, Fortnow, Rubinfeld, Smith, and White [5], with a flurry of works over the past two decades addressing a range of hypothesis testing tasks, under various models of distribution testing (see, e.g., [34, 10, 11] and [29, Chapter 11] for surveys and expositions).

Two of the flagship and most fundamental tasks in distribution testing are *identity testing* (and its special case, *uniformity testing*), and *closeness testing* (also known as *equivalence* testing): in the first, an algorithm is given independent samples from an (unknown) discrete

© Clément L. Canonne, Yash Pote, Jonathan Scarlett, and Joy Qiping Yang;
licensed under Creative Commons License CC-BY 4.0

probability distribution, and must decide whether the distribution is equal to some known reference distribution, or significantly deviates from it in a meaningful notion of distance (typically total variation distance). In the second, the algorithm has access to samples from *two* unknown distributions, and similarly must decide whether they are equal, or significantly far from each other. Here, "significantly far" is quantified by an input parameter $\varepsilon > 0$, and the algorithm is required to be correct with high probability $1 - \delta$,[1] while minimizing the *sample complexity*, that is, the total number of samples taken from the distribution(s) in the worst case as a function of the domain size n, distance parameter ε, and error probability δ.

The sample complexity of these two flagship tasks is by now well understood, with several time-efficient algorithms known to achieve the optimal sample complexity, as a function of all parameters: namely,

$$\Theta\left(\frac{\sqrt{n\log(1/\delta)} + \log(1/\delta)}{\varepsilon^2}\right)$$

for identity testing [32, 1, 28, 27], and

$$\Theta\left(\frac{n^{2/3}\log^{1/3}(1/\delta) + \log(1/\delta)}{\varepsilon^{4/3}} + \frac{\sqrt{n\log(1/\delta)} + \log(1/\delta)}{\varepsilon^2}\right)$$

for closeness testing [5, 17, 28, 26]. However, in spite of these impressive advances, these sample complexities can remain prohibitively large for many combinations of (n, ε), notably for small ε corresponding to distributions that may be very close to one another. Given that their sample complexities are constant-factor *optimal*, there would seem to be little room for improvement, yet two possible avenues remain. The first is to *relax the guarantees*: total variation is known to be a very stringent measure of distance, and testing with respect to a more permissive distance measure may enable significant sample complexity savings, while still being "good enough" for the application at hand. This direction was partially explored in, for instance, [24]. The second avenue, particularly well suited to high-dimensional data, is to *make additional structural assumptions* on the distribution: high-dimensional data is seldom fully unstructured, and one can usually leverage domain knowledge to make modelling assumptions, for instance that the probability distributions exhibit a low-degree Bayesian network structure, or are consistent with an Ising model. This line of work, in the context of distribution testing, was initiated in [12, 23].

Our work makes progress on both of these fronts. We first define a relaxation of closeness testing, which we term *entropy equivalence testing*, where one is asked to detect if the two unknown distributions have significantly different (Shannon) entropy:[2]

> **Entropy Equivalence Testing (EET)**: Given sample access to distributions p, q over a known domain of size n and parameters $\varepsilon \in (0, 1/2]$ and $\delta \in (0, 1]$, distinguish, with probability at least $1 - \delta$, between the following two cases: (1) $p = q$, and (2) $|H(p) - H(q)| \geq \varepsilon$.

[1] One typically sets δ to be a small constant, e.g., $\delta = 1/10$: this is without much loss of generality, as one can amplify this by repetition to any target $1 - \delta$, at the cost of an $O(\log(1/\delta))$ factor in sample and time complexity.

[2] Recall that the (Shannon) entropy of a probability distribution p over a discrete domain $\mathcal{X}$ is defined as $H(p) = \sum_{x\in\mathcal{X}} p(x)\log\frac{1}{p(x)}$.

We note that the "single-distribution" version of this task, when q is fully known to the algorithm, was recently introduced [15], under the name Entropy Identity Testing (EIT). [15] then leveraged their algorithm for EIT to obtain an improved identity testing algorithm for "standard" identity testing (i.e., in total variation distance) of *bounded-in-degree Bayes nets*, establishing a connection between the two aforementioned avenues of research. We build upon and deepen this connection to obtain, from our algorithm for EET, the first non-trivial *closeness* testing algorithm for Bayes nets.

1.1 Our results

We now describe our results in more detail. Our first contribution is an efficient algorithm for the EET task defined above (where we implicitly set the probability of failure δ to be a small constant):

▶ **Theorem 1.** *There exists an algorithm for the Entropy Equivalence Testing problem with sample complexity*

$$\tilde{O}\left(\frac{n^{3/4}}{\varepsilon}\right) + O\left(\frac{\log^2(n/\varepsilon)}{\varepsilon^2}\right)$$

and running time near-linear in the number of samples.

Combining Theorem 1 and the "baseline tester" discussed below (see Section 1.2) which follows from a reduction from entropy difference to total variation distance and has sample complexity $\tilde{O}\left(\max\left(\frac{n^{2/3}}{\varepsilon^{4/3}}, \frac{\sqrt{n}}{\varepsilon^2}\right)\right)$, one gets, after simplifying the resulting sample complexity, our main result:

▶ **Corollary 2** (Main testing result)**.** *There exists an algorithm for the Entropy Equivalence Testing problem with sample complexity*

$$\tilde{O}\left(\min\left\{\frac{n^{3/4}}{\varepsilon}, \frac{n^{2/3}}{\varepsilon^{4/3}}\right\}\right) + O\left(\frac{\log^2(n/\varepsilon)}{\varepsilon^2}\right)$$

and running time near-linear in the number of samples.

We complement this positive result with an information-theoretic lower bound, showing that our algorithm has nearly optimal sample complexity:

▶ **Theorem 3** (Lower bound)**.** *Any algorithm for the Entropy Equivalence Testing problem must draw*

$$\widetilde{\Omega}\left(\min\left\{\frac{n^{3/4}}{\varepsilon}, \frac{n^{2/3}}{\varepsilon^{4/3}}\right\}\right) + \Omega\left(\frac{\log^2 n}{\varepsilon^2}\right)$$

total samples from p and q.

Finally, we show how to leverage our algorithm for EET to obtain a closeness testing algorithm for bounded in-degree Bayes nets: we emphasize that, to the best of our knowledge, this is the *first* testing algorithm for this problem, besides the "naïve" (and computationally inefficient) testing-by-learning approach that fully learns the two unknown distributions with $\tilde{O}(2^d n/\varepsilon^2)$ samples, in time $2^{\Omega(n)}$:

▶ **Theorem 4.** *There exists an algorithm for closeness testing (in total variation distance) of in-degree d Bayesian networks with sample complexity*

$$s = \tilde{O}\left(\min\left\{\frac{2^{3d/4}n}{\varepsilon^2}, \frac{2^{2d/3}n^{4/3}}{\varepsilon^{8/3}}\right\} + \frac{n^2}{\varepsilon^4}\right),$$

and running time $\tilde{O}(n^{d+1}s)$.

Interestingly, the techniques underlying Theorem 4 can also be easily adapted to obtain an algorithm for identity testing of Bayes nets, improving upon the work of [15]: see Corollary 18.

1.2 Technical overview

We now provide an overview of the techniques and main ideas underlying our results.

1.2.1 Upper bound for EET

A bound of $\tilde{O}\left(\max\left\{\frac{n^{2/3}}{\varepsilon^{4/3}}, \frac{\sqrt{n}}{\varepsilon^2}\right\}\right)$ readily follows from a standard inequality, allowing one to reduce the entropy difference problem to closeness testing in TV: namely, when $d_{\mathrm{TV}}(p,q) \leqslant \frac{1}{2}$, then $|H(p) - H(q)| \leqslant d_{\mathrm{TV}}(p,q) \log \frac{n}{d_{\mathrm{TV}}(p,q)}$ [21, Lemma 2.7]. Another straightforward upper bound is obtained by estimating the entropy, a harder task, which takes $\Theta\left(\frac{n}{\varepsilon \cdot \log n} + \frac{\log^2 n}{\varepsilon^2}\right)$ samples [42]. This hints at the possibility of getting a better bound of $O(n^c/\varepsilon + \log^2 n/\varepsilon^2)$ for some absolute constant $c < 1$. Surprisingly, the final upper bound of $\tilde{O}\left(\min\left\{\frac{n^{2/3}}{\varepsilon^{4/3}}, \frac{n^{3/4}}{\varepsilon}\right\} + \frac{1}{\varepsilon^2}\right)$ is related to the squared Hellinger distance (and not the total variation). This is a result of considering two testers that can solve EET independently, and choosing the one with lower sample complexity. The main contribution of our work is providing an alternative tester to the TV reduction tester above.

An $\tilde{O}(\frac{n^{3/4}}{\varepsilon} + \frac{1}{\varepsilon^2})$ tester. Our analysis starts with a reduction to testing both farness (vs. equality) in Hellinger distance and a cross-entropy difference term (with respect to the uniform mixture) via some algebraic manipulations (see Lemma 9). The problem thus becomes testing whether Hellinger distance is large (which we know costs $O(\min\{\frac{n^{3/4}}{\varepsilon}, \frac{n^{2/3}}{\varepsilon^{4/3}}\}) \leqslant O(\frac{n^{3/4}}{\varepsilon})$) and testing whether the cross-entropy term is large (since one of them will be large if the entropy difference is large). A natural idea is of course to employ a plug-in estimator for the cross entropy difference (see (1)). Unfortunately, this empirical estimator has an inherent[3] bias of the form $\sum_{i \in [n]} \frac{|p_i - q_i|}{m(p_i+q_i)}$, where m is the (equal) number of samples we take from p and q to compute the estimate. Our argument to deal with the bias is as follows: If the expectation of the plug-in estimator has *low bias* (at most $O(\varepsilon)$), then we can rely on its estimate alone. On the other hand, if it has *high bias* (at least $\Omega(\varepsilon)$), we hope to catch it and reject (i.e., declare that $|H(p) - H(q)| > \epsilon$) early.

Low mass regime: Reduction to TV. One main challenge to designing this test is that the bias is impossible to test without a lower bound on $p_i + q_i$ (the variance of estimating the bias could be huge). The key to passing this hurdle is an idea inspired by a different algorithm from [28], whose analysis implies that for the elements $i \in [n]$ where q_i (or p_i) is small, we can perform tests more efficiently (in TV) via access to its conditional distribution (as efficiently as $\tilde{O}(n^{3/4}/\varepsilon)$), with the intuition being: The TV distance between the conditional distributions is actually larger (inverse of its mass size $\tilde{\Omega}(\varepsilon/q(T))$, where T is the set of small-mass elements). Via a reduction to TV from cross-entropy difference, we can also test farness on these elements with $\tilde{O}(n^{3/4}/\varepsilon)$ samples.

[3] In the works of [42], their plug-in estimator also exhibits similar bias in the analysis, and they resort to machinery from polynomial approximation to reduce the bias (which will increase the variance the higher the degree of the polynomial).

Detecting bias in the high mass regime. Capitalizing on this, we will consider exclusively elements with large mass: $p_i \geqslant \tilde{\Omega}(\varepsilon/n^{3/4})$. With a bound on $m(p_i + q_i) \geqslant \Omega(1)$ (since we take $m = \tilde{O}(\frac{n^{3/4}}{\varepsilon})$), this reduces to testing farness in TV distance, which can be performed with $O(\sqrt{n}/\varepsilon^2)$ samples (to bound the bias term by $O(\varepsilon)$), rather than the $O(n^c/\varepsilon)$ for some $c < 1$ that we set out for. Luckily our problem can be easier (depending on relation between n and ε) than testing in TV (see Lemma 13). Intuitively, this is related to testing in the instance-optimal setting [40, 28] – one can test with sample complexity as a function of p, q (for our case p, q has exclusively large probability masses), rather than as a function of the domain size (the worst case over all possible p, q). Interestingly, to detect the bias, we are using the same test statistic (with different threshold) first used in [17] to test in TV distance and subsequently in [24] to test in Hellinger distance.

Had we not made a "reject" decision after the bias check in the high mass regime (nor rejected due to its low mass cross-entropy difference), it would suggest that the bias on the high mass regime is at most $O(\varepsilon)$ and therefore, the estimator for $\sum_i (p_i - q_i) \log((p_i + q_i)/2)$ is reliable: A standard analysis of expectation and variance followed by an application of Chebyshev's inequality then yields Theorem 1.

1.2.2 Lower bound for EET

The lower bound is a combination of $\tilde{\Omega}(\min\{\frac{n^{2/3}}{\varepsilon^{4/3}}, \frac{n^{3/4}}{\varepsilon}\})$ and $\Omega(\frac{\log^2 n}{\varepsilon^2})$. The former is derived via a reduction from testing mutual information to entropy difference. The task of testing mutual information (MI) is as follows: if $I(X : Y) = 0$, then accept with probability 0.99; If $I(X : Y) \geqslant \varepsilon$, then reject with probability 0.99. Given the close relation between mutual information and entropy: $I(X : Y) = H(X) + H(Y) - H(X, Y)$, one can reduce the problem of MI testing to EET by treating p as the joint distribution of X, Y and q as the product of X and Y (we can simulate sample access to the product of X and Y with two samples from the joint distribution). Therefore, the existing lower bound from [37, Corollary 6.2] holds for our problem. The latter lower bound of $\Omega(\frac{\log^2 n}{\varepsilon^2})$ comes from [15, Lemma 2.7], given the reduction from identity testing to equivalence testing.

1.2.3 Application to equivalence testing of Bayes nets

As laid out by [15], one can utilize testing entropy difference as a subroutine to testing Bayes nets. The key of the analysis is derived via the celebrated Chow-Liu decomposition [20]: If two Bayes nets p and q have similar local entropies, then they must share approximately similar structures (their corresponding DAGs are close in some sense; see [15, Lemma 3.4]). Once we confirm that they share a similar structure, the problem boils down to checking local (every subset of random variables of size at most $d + 1$) KL-divergences. While KL-divergences are known to be untestable [24, Theorem 7] in the extreme cases when $q(x) = 0$, there is hope if one can obtain a lower bound on $q(x)$. In fact, we can reduce testing in KL to testing in Hellinger distance with an $O\left(\max_x \log \frac{p(x)}{q(x)}\right)$ factor overhead using [37, Lemma 3.1]. To get a lower bound on $q(x)$, we use a standard technique of testing the mixture of q and the uniform distribution.

1.3 Related works

Estimating entropy. Entropy estimation is a well-studied problem in both the information theory [41, 42, 31] and property testing [39, 3] communities, with the near-optimal sample complexity $\Theta(\frac{n}{\varepsilon \log n} + \frac{\log^2 n}{\varepsilon^2})$ being eventually established by [42, 31] (for discrete distributions).

As we are considering the testing problem, our upper bound is less costly compared to its learning (estimation) variant. Moreover, other variants of entropy have been studied, such as Rényi entropy [2]. The more general problem of estimating functionals of distributions was also considered by [31].

Identity testing and closeness testing. The standard formulation of identity testing asks the following problem: Given a full description of a hypothesis q, how many i.i.d. samples are needed from p to distinguish between $p = q$ and $d_{\mathrm{TV}}(p, q) \geqslant \varepsilon$, where p and q are both distributions over support of size $[n]$? This problem is well-known to have a sample complexity of $\Theta\left(\sqrt{n}/\varepsilon^2\right)$, a much lower cost compared to fully learning every parameter of p, which takes $\Theta(n/\varepsilon^2)$ samples. A similar problem can be asked if one is instead only provided sample access to p and q, both unknown. This is the closeness testing problem (in TV) and it is known that a $\Theta\left(\max\{\frac{\sqrt{n}}{\varepsilon^2}, \frac{n^{2/3}}{\varepsilon^{4/3}}\}\right)$ sample complexity is necessary and sufficient, making this a slightly harder problem than identity testing while still saving on samples compared to fully learning p and q. Our problem differs from the standard formulation by testing in the entropy difference rather than in TV distance. Most relevant to our work is the work of [15], which solves the problem of entropy identity testing (up to polylogarithmic factors) with a sample complexity of $\tilde{\Theta}(\frac{\sqrt{n}}{\varepsilon} + \frac{1}{\varepsilon^2})$. Another relevant work is that of [24], which asks the question of testing in different distances, e.g., distinguishing between $d_{\mathrm{KL}}(p\|q) \leqslant \varepsilon^2/100$ and $d_{\mathrm{H}}(p, q) \geqslant \varepsilon$ for identity testing (sample access to p and the full description of q). Interestingly, as shown in the same work, equivalence testing in squared Hellinger distance, i.e., distinguishing between $p = q$ and $d_H^2(p, q) \geqslant \varepsilon$ has a similar cost of $\Theta\left(\min\{\frac{n^{2/3}}{\varepsilon^{4/3}}, \frac{n^{3/4}}{\varepsilon}\}\right)$ (though perhaps not so surprising, as we are reducing to testing in squared Hellinger square in one of the cases we handle). Our work adds to their results for equivalence testing in entropy difference.

Testing mutual information. Testing mutual information (MI) asks the following question: Given sample access to a (discrete) distribution p over two random variables X, Y with a domain of size $|\Sigma_X|$ and $|\Sigma_Y|$ respectively, how many samples do we need from p to distinguish between $I(X : Y) = 0$ and $I(X : Y) \geqslant \varepsilon$? The study of this problem was initiated by [9] to solve the problem of learning a special variant of Bayes net (trees). While their MI tester is sufficient for learning trees-structured distributions near-optimally, their upper bound is loose in terms of dependence on $|\Sigma_X|$ and $|\Sigma_Y|$: $O(\frac{|\Sigma_X|\cdot|\Sigma_Y|}{\varepsilon})$. More recently, [37] gave a near-optimal bound of $\tilde{\Theta}(\min\{\frac{|\Sigma_X|^{3/4}|\Sigma_Y|^{1/4}}{\varepsilon}, \frac{|\Sigma_X|^{2/3}|\Sigma_Y|^{1/3}}{\varepsilon^{4/3}}\})$ (assuming $|\Sigma_X| \geqslant |\Sigma_Y|$). We note that this bound bears similarity (e.g., setting $|\Sigma_Y| = O(1)$) to ours due to the close relation between MI and entropy: $I(X : Y) = H(X) + H(Y) - H(X, Y)$. Indeed, MI testing can be reduced to EET naturally (as outlined above). Before these works, [4, 13] studied the problem of (conditional) independence testing in the TV distance.

Testing Bayesian networks. A Bayesian network (Bayes net) is a high-dimensional distribution over n random variables that factorizes into n conditional probabilities: $p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i \mid \Pi_i)$, where $\Pi_i \in [n]$ is called the parent set of X_i. This factorization is associated with a directed acyclic graph (DAG) $G = (V, E)$ ($V = \{X_1, \ldots, X_n\}$), and if $X_j \in \Pi_i$, then the edge $(X_j, X_i) \in E$ (with X_j being one of X_i's parents). A distribution p is called a Bayes net if its joint probability density function factorizes according to G and p is called Markov with respect to G. Learning Bayesian networks from samples is a fundamental problem in learning theory, and there is a plethora of evidence that this problem is hard computationally [18, 22, 7]. On the other hand, time-efficient algorithms (polynomial in n, the number of random variables) do exist in some special cases [19, 8, 9].

Identity testing and closeness testing of Bayes nets were systematically studied by [12], though their stronger upper bounds only consider relaxed version of Bayes nets where both p and q are required be *balanced* and *non-degenerate*. While they also provide a more general upper bound (without assuming balancedness and non-degeneracy for Bayes nets), they still require that the node ordering be the same between p and q, a non-trivial assumption. Our upper bound does not make any assumption other than the bounded in-degree d, which is a natural and necessary parameter to consider, since testing unbounded degree Bayes nets is exponentially hard: $\Omega(\frac{2^{n/2}}{\varepsilon^2})$ (e.g., see [6, Theorem B.1]). The strongest upper bound that we are aware of is by [25], which can solve the identity testing[4] problem in $\tilde{O}(\frac{2^{d/2}n}{\varepsilon^2})$ samples,[5] under the assumption that p and q have the same structure (Markov with respect to the same DAG). For equivalence testing, under the same assumption, they give a bound of $\tilde{O}(\min\{\frac{2^{3d/4}n}{\varepsilon^2}, \frac{2^{2d/3}n^{4/3}}{\varepsilon^{8/3}}\})$.[6] Our results match their bounds if $d \gg \log(n/\varepsilon)$, and our algorithm will work without the extra assumption that p and q share the same structure. A lower bound of $\Omega(\frac{2^{d/2}n}{\varepsilon^2})$ when $d \leqslant \log n$ [6, Theorem 4.1] holds for identity testing of Bayes nets and thus also for its closeness testing variant (our setting).

1.4 Preliminaries

Define the factorial moment by $(x)_m := \frac{x!}{(x-m)!}$, which gives an unbiased estimator for the monomials of the Poisson mean: namely, if $X \sim \mathrm{Poi}(\lambda)$, then $\mathbb{E}[(X)_m] = \lambda^m$ for all $m \geqslant 0$. More generally, we have the following fact, whose proof we provide in Appendix A for completeness:

▶ **Fact 5.** *Let X be a Poisson random variable with parameter λ, and $f\colon \mathbb{R} \to \mathbb{R}$ be any function, then, we have that $\mathbb{E}[(X)_m \cdot f(X)] = \lambda^m \cdot \mathbb{E}[f(X+m)]$. In particular, choosing $f \equiv 1$, we have that $\mathbb{E}[(X)_m] = \lambda^m$ for all $m \geqslant 0$.*

▶ **Fact 6** (Folklore). *For any $0 < \alpha \leqslant 1/2$ and $\varepsilon \in (0,1]$, testing whether the bias of a coin is at most α or at least $\alpha(1+\varepsilon)$, with probability at least $1-\delta$, can be done with $n = O\left(\frac{1}{\alpha\varepsilon^2}\log\frac{1}{\delta}\right)$ samples.*

Distances and norms. We will use a number of standard distance measures for two distributions p, q over $[n]$, and various known inequalities between them, e.g., see [36]. The *squared Hellinger distance* is

$$d_H^2(p,q) := \frac{1}{2}\sum_{i\in[n]}\left(\sqrt{p_i}-\sqrt{q_i}\right)^2$$

The *total variation distance* is $d_{\mathrm{TV}}(p,q) := \frac{1}{2}\|p-q\|_1 = \frac{1}{2}\sum_{i\in[n]}|p_i - q_i|$. The *$\ell_2$ distance* is $\|p-q\|_2 := \left(\sum_{i\in[n]}(p_i-q_i)^2\right)^{1/2}$, and the *$\chi^2$-divergence* of p from q is $d_{\chi^2}(p,q) := \sum_{i\in[n]}(p_i-q_i)^2/q_i$. These satisfy the chain of inequalities

$$d_{\mathrm{TV}}(p,q) \leqslant \sqrt{2}\, d_H(p,q) \qquad \text{and} \qquad d_H^2(p,q) \leqslant d_{\mathrm{TV}}(p,q) \leqslant \sqrt{d_{\chi^2}(p,q)},$$

and the equivalence $\frac{1}{2}\sum_i (p_i-q_i)^2/(p_i+q_i) = \Theta(d_H^2(p,q))$. The *Kullback–Leibler (KL) divergence* is $d_{\mathrm{KL}}(p\|q) := \sum_i p_i \log(p_i/q_i)$, which satisfies $d_{\mathrm{KL}}(p\|q) \leqslant d_{\chi^2}(p,q)$ and Pinsker's inequality $d_{\mathrm{TV}}(p,q) \leqslant \sqrt{d_{\mathrm{KL}}(p\|q)/2}$.

[4] Note that their setting of identity testing actually corresponds to equivalence testing in our setting.
[5] Implicitly, using the Hellinger testing result of [24, Theorem 1].
[6] Implicitly, using the optimal Hellinger equivalence test [24].

Notation. In what follows, for a given integer $n \geqslant 1$, we denote by $[n]$ the set $[n] = \{1, 2, \ldots, n\}$, and by U_n the uniform distribution on $[n]$. When clear from context, we will omit the subscript, and write U instead of U_n. Given a probability distribution p over $[n]$ and a set $S \subseteq [n]$ such that $p(S) > 0$ we write $p(\cdot \mid S)$ for the conditional distribution of p on S. For the high dimensional (Bayes net) section, we use p_T to denote the marginalization of p over T, where T is a subset of $\{X_1, \ldots, X_n\}$, e.g., p_{X_1,X_2} denotes the marginal distribution of (X_1, X_2). We also use Π_i to denote the parent set of the random variable X_i in a Bayes net.

Throughout the paper, we use log for the natural logarithm, and adopt the standard asymptotic notation $O(\cdot), \Omega(\cdot), \Theta(\cdot)$, as well as the closely related $\tilde{O}(\cdot), \tilde{\Omega}(\cdot), \tilde{\Theta}(\cdot)$, which hide polylogarithmic factors in their argument.

2 An $\tilde{O}(\frac{n^{3/4}}{\varepsilon} + \frac{\log^2(n/\varepsilon)}{\varepsilon^2})$ Upper Bound for EET

In this section, we establish our main theorem, Theorem 1. We begin with a simple algorithmic "trick" that allows us to reduce the task to one where both p, q put at least some minimum probability $\eta > 0$ on every element of the domain:

▶ **Lemma 7.** *Let p, q be distributions over $[n]$, and set $\eta := \varepsilon/(4\log(n/\varepsilon))$. Let $\tilde{p} := (1-\eta)p + \eta U$ and likewise for $\tilde{q}$. Then $\tilde{p}_i, \tilde{q}_i \geqslant \varepsilon/(4n\log(n/\varepsilon))$ for every $i \in [n]$, and*

$$||H(\tilde{p}) - H(\tilde{q})| - |H(p) - H(q)|| \leqslant \varepsilon/2.$$

Moreover, one sample from $\tilde{p}$ (resp. $\tilde{q}$) can be simulated from one sample from p (resp. q).

Proof. Since $d_{\mathrm{TV}}(p, \tilde{p}) \leqslant \frac{\eta}{2} = \frac{\varepsilon}{8\log(n/\varepsilon)}$, by invoking the bound on entropy difference with respect to TV distance in [21, Lemma 2.7], we get

$$|H(p) - H(\tilde{p})| \leqslant \frac{\varepsilon}{8\log(n/\varepsilon)} \log\left(\frac{n \cdot 8\log(n/\varepsilon)}{\varepsilon}\right) = \varepsilon \cdot \frac{\log(n/\varepsilon) + \log(8\log(n/\varepsilon))}{8\log(n/\varepsilon)} \leqslant \varepsilon/4.$$

Similarly, we have $|H(q) - H(\tilde{q})| \leqslant \varepsilon/4$. By the triangle inequality, we then have

$$|H(\tilde{p}) - H(\tilde{q})| \leqslant |H(p) - H(\tilde{p})| + |H(q) - H(\tilde{q})| + |H(p) - H(q)| \leqslant \varepsilon/2 + |H(p) - H(q)|,$$

and

$$|H(p) - H(q)| \leqslant |H(\tilde{p}) - H(\tilde{q})| + |H(p) - H(\tilde{p})| + |H(q) - H(\tilde{q})| \leqslant \varepsilon/2 + |H(\tilde{p}) - H(\tilde{q})|,$$

establishing the lemma. ◀

Of course, if $p = q$, then $\tilde{p} = \tilde{q}$; Lemma 7 guarantees that if $|H(p) - H(q)| \geqslant \varepsilon$, then $|H(\tilde{p}) - H(\tilde{q})| \geqslant \Omega(\varepsilon)$. That is, $\tilde{p}, \tilde{q}$ satisfy the promise with the same ε (up to constants) whenever p, q do, so we can henceforth assume that $\min_{i\in[n]} p_i, \min_{i\in[n]} q_i \geqslant \frac{\varepsilon}{4n\log(n/\varepsilon)}$. With this in hand, we are ready to establish our main theorem, restated, as follows:

▶ **Theorem 8** (Theorem 1, restated)**.** *There exists an algorithm (Algorithm 1) which, given sample access to two unknown probability distributions p, q over a known domain $[n]$, as well as parameter $\varepsilon \in (0, 1/2]$, takes*

$$O\left(\frac{n^{3/4}\log(n/\varepsilon)\cdot\log n}{\varepsilon} + \frac{\log^2(n/\varepsilon)}{\varepsilon^2}\right)$$

samples from both p and q, and has the following guarantees:

Algorithm 1 Entropy equivalence testing (EET).

Require: Sample access to p and q, both over $[n]$; accuracy parameter $\varepsilon \in (0,1]$.

Run a Hellinger closeness tester on p, q with $O\left(\frac{n^{3/4}}{\varepsilon}\right)$ samples.
if $d_H^2(p,q) \geqslant \Omega(\varepsilon)$ **then return** reject
Set $m_1 := \tilde{O}(n^{3/4}/\varepsilon)$. Take m_1 samples each from p, q; let X_i, Y_i be the empirical counts.
$S \leftarrow \left\{ i \in [n] : X_i + Y_i \geqslant \Omega(m_1 \cdot \varepsilon/(n^{3/4}\log(n/\varepsilon))) \right\}$ ▷ heavy elements
Set $m_2 := \tilde{O}(1/\varepsilon)$. Using m_2 samples, test whether $p(\bar{S}) \geqslant \tilde{\Omega}(\varepsilon)$ and $q(\bar{S}) \geqslant \tilde{\Omega}(\varepsilon)$.
if only one of the preceding inequalities hold **then** ▷ p and q differ on $\bar{S}$.
    return reject
if both preceding inequalities hold **then** ▷ low-mass regime on $\bar{S}$
    Set $m_3 := O(\log^2(n/\varepsilon)/\varepsilon^2)$. Using m_3 samples, estimate $|p(\bar{S}) - q(\bar{S})|$.
    if $|p(\bar{S}) - q(\bar{S})| \geqslant \tilde{\Omega}(\varepsilon)$ **then return** reject
    Via rejection sampling on $\bar{S}$, run a TV closeness tester on $p_{\bar{S}}, q_{\bar{S}}$ at threshold $\tilde{\Omega}(\varepsilon/q(\bar{S}))$ using $\tilde{O}(n^{3/4}/\varepsilon)$ samples.
    if the conditional tester rejects **then**
        return reject
Set $m_4 := \tilde{O}\left(n^{3/4}/\varepsilon + \log^2(n/\varepsilon)/\varepsilon^2\right)$ and $s := \Omega(n^{3/4}\log(n/\varepsilon)/\varepsilon)$.
Using s samples, compute $T \leftarrow \sum_{i \in S} \frac{(X_i - Y_i)^2 - (X_i + Y_i)}{X_i + Y_i}$. ▷ bias check
if $T \geqslant \Omega(\sqrt{n})$ **then**
    return reject
Using $O(\log^2(n/\varepsilon)/\varepsilon^2)$ samples, test both $|p(S) - q(S)| \geqslant \Omega\left(\frac{\varepsilon}{\log(n/\varepsilon)}\right)$ and $\|p - q\|_2^2 \geqslant \Omega\left(\frac{\varepsilon^2}{\log^2(n/\varepsilon)}\right)$.
if either preceding inequality holds **then**
    return reject
Take m_4 samples; let X_i, Y_i be the empirical counts on S. Compute

$$Z \leftarrow \sum_{i \in S} \frac{X_i - Y_i}{m_4} \log \frac{1}{X_i + Y_i}.$$

if $|Z| \geqslant \Omega(\varepsilon)$ **then return** reject
else return accept

- *If $p = q$, then the algorithm outputs* accept *with probability at least $9/10$;*
- *If $|H(p) - H(q)| \geqslant \varepsilon$, then the algorithm outputs* reject *with probability at least $9/10$.*

Further, the algorithm runs in time $O(N \log N)$, where N is the total number of samples.

The remainder of this section is dedicated to proving this theorem, which we do via a series of lemmas.

The first lemma relates a difference in entropy to a difference in (squared) Hellinger distance and a cross entropy term with respect to a mixture:

▶ **Lemma 9.** *For any two distributions p, q over $[n]$, we have*

$$|H(p) - H(q)| \leqslant \Theta(d_H^2(p,q)) + \left| \sum_{i=1}^{n} (p_i - q_i) \log \frac{1}{(p_i + q_i)/2} \right|.$$

Proof. Letting $r \coloneqq \frac{p+q}{2}$, we can write

$$\begin{aligned}
&H(p) - H(q) \\
&= \sum_i \left(p_i \log \frac{1}{p_i} - q_i \log \frac{1}{q_i} \right) \\
&= \sum_i \left(-p_i \log \frac{p_i}{(p_i+q_i)/2} + p_i \log \frac{1}{(p_i+q_i)/2} + q_i \log \frac{q_i}{(p_i+q_i)/2} - q_i \log \frac{1}{(p_i+q_i)/2} \right) \\
&= d_{\mathrm{KL}}\left(q \,\|\, r\right) - d_{\mathrm{KL}}\left(p \| r\right) + \sum_i \left(p_i \log \frac{1}{(p_i+q_i)/2} - q_i \log \frac{1}{(p_i+q_i)/2} \right) \\
&= d_{\mathrm{KL}}\left(q \,\|\, r\right) - d_{\mathrm{KL}}\left(p \| r\right) + \sum_i (p_i - q_i) \log \frac{1}{(p_i+q_i)/2}.
\end{aligned}$$

We then observe that

$$d_{\mathrm{KL}}\left(q \,\|\, r\right) \leqslant d_{\chi^2}\left(q, r\right) = \sum_i \frac{\left(q_i - \frac{p_i+q_i}{2}\right)^2}{\frac{p_i+q_i}{2}} = \sum_i \frac{1}{2} \frac{(p_i - q_i)^2}{p_i + q_i} = \Theta(d_H^2(p,q)),$$

where the last equality comes from the equivalence of between Hellinger distance and Triangle distance (e.g., see [24, Proposition 3]).[7] Similarly, we have that $d_{\mathrm{KL}}\left(p \,\|\, r\right) \leqslant d_{\chi^2}\left(p, r\right) = \Theta(d_H^2(p,q))$, and we conclude that

$$\begin{aligned}
|H(p) - H(q)| &\leqslant |d_{\mathrm{KL}}\left(q \,\|\, r\right)| + |d_{\mathrm{KL}}\left(p \,\|\, r\right)| + \left| \sum_i (p_i - q_i) \log \frac{1}{(p_i+q_i)/2} \right| \\
&\leqslant \Theta(d_H^2(p,q)) + \left| \sum_i (p_i - q_i) \log \frac{1}{(p_i+q_i)/2} \right|,
\end{aligned}$$

as claimed. ◀

Lemma 9 allows us to split the entropy difference $|H(p) - H(q)|$ into a Hellinger part and a term $|\sum_i (p_i - q_i) \log(1/((p_i+q_i)/2))|$, so that an entropy difference ε implies at least one of these two terms must be $\varepsilon/2$: thus, it suffices to test both, and reject if, and only if, at least one of them is detected to be $\Omega(\varepsilon)$. (Clearly, if $p = q$, both the Hellinger and KL-like terms are zero.)

The Hellinger term can be tested with $\widetilde{O}(n^{3/4}/\varepsilon)$ samples using a Hellinger closeness tester (e.g., [28, Proposition 2.15]): if $d_H^2(p,q) \geqslant \Omega(\varepsilon)$, this tester will allow us to reject with high probability. For the rest of the subsection we therefore can assume $d_H^2(p,q) \leqslant O(\varepsilon)$, and the task reduces to testing the second, "KL-like" term,

$$\Lambda(p,q) \coloneqq \left| \sum_i (p_i - q_i) \log \frac{1}{(p_i+q_i)/2} \right|, \tag{1}$$

and specifically to distinguish $\Lambda(p,q) = 0$ from $\Lambda(p,q) \geqslant \frac{\varepsilon}{2}$.

To do so, the idea is to handle separately the set S^* of "heavy" elements, on which either p or q puts relatively high probability (specifically, at least some threshold $\tau \coloneqq \tilde{\Omega}(\varepsilon/n^{3/4})$), and the remaining set $\bar{S^*} = [n] \setminus S^*$ of "light" elements (which have low probability under

[7] Note that the induced KL terms have been studied in information theory (called Jensen-Shannon divergence; see e.g., [33, Chapter 7] and more systematically in [38], where they actually show that the Jensen-Shannon divergence and the Triangle distance are equivalent up to constant factors).

both p and q). Of course, we do not know this set S^*, as both p, q are unknown; however, by taking $\tilde{O}(1/\tau) = \tilde{O}(n^{3/4}/\varepsilon)$ samples (which is inconsequential, as we will pay this sample complexity for the Hellinger test anyway), we can obtain a set S which is a good proxy for S^*. To make this formal, define the following two sets:

$$S_1 := \left\{ i \in [n] \mid p_i + q_i \geqslant C_1 \cdot \frac{\varepsilon}{n^{3/4}\log(n/\varepsilon)} \right\}.$$
$$S_2 := \left\{ i \in [n] \mid p_i + q_i \geqslant C_2 \cdot \frac{\varepsilon}{n^{3/4}\log(n/\varepsilon)} \right\},$$

where $C_2 \geqslant C_1 > 0$ are two absolute constants. Our focus is to identify a set $S \subseteq [n]$ such that $S_2 \subseteq S \subseteq S_1$.

▶ **Lemma 10.** *Using $O\left(\frac{n^{3/4}\log(n/\varepsilon)\log n}{\varepsilon}\right)$ samples from p and q, one can efficiently return a set S satisfying $S_2 \subseteq S \subseteq S_1$ with probability at least $\frac{99}{100}$.*

Proof. By Fact 6, choosing $2C_1 \leqslant C_2$, as we have sample access to the uniform mixture $\frac{p+q}{2}$, we can distinguish $\frac{p_i+q_i}{2} \geqslant C_2 \cdot \frac{\varepsilon}{2n^{3/4}\log(n/\varepsilon)}$ and $\frac{p_i+q_i}{2} \leqslant C_1 \cdot \frac{\varepsilon}{2n^{3/4}\log(n/\varepsilon)}$ with probability $1 - \frac{1}{100n}$ costing $O\left(\frac{n^{3/4}\log(n/\varepsilon)\cdot\log n}{\varepsilon}\right)$ samples from p and q. Therefore, running the test over the support $[n]$ (using the same set of samples to run this test for all i's) and adding each $i \in [n]$ to the set S if we detect $p_i + q_i \geqslant C_2 \cdot \frac{\varepsilon}{n^{3/4}\log(n/\varepsilon)}$, by a union bound, the algorithm will be simultaneously correct on all $i \in [n]$ with probability at least $\frac{99}{100}$. ◀

To summarize, once we have identified such a set S, given that

$$\underbrace{\Lambda(p,q)}_{0 \text{ or } \geqslant \frac{\varepsilon}{2}} \leqslant \underbrace{\left|\sum_{i\in\overline{S}}(p_i - q_i)\cdot\log\frac{1}{p_i+q_i}\right|}_{=:\Lambda_{\overline{S}}(p,q)} + \underbrace{\left|\sum_{i\in S}(p_i - q_i)\cdot\log\frac{1}{p_i+q_i}\right|}_{=:\Lambda_S(p,q)}$$

it suffices to separately test (1) $\Lambda_{\overline{S}}(p,q) = 0$ vs. $\Lambda_{\overline{S}}(p,q) \geqslant \varepsilon/4$, and (2) $\Lambda_S(p,q) = 0$ vs. $\Lambda_S(p,q) \geqslant \varepsilon/4$; we do so below, assuming that our set S satisfies $S_2 \subseteq S$.[8]

Test 1: $\Lambda_{\overline{S}}(p,q) = 0$ vs. $\Lambda_{\overline{S}}(p,q) \geqslant \varepsilon$

Inspired by the ideas of [28], we can then check if the distributions are far *in TV distance* in the low-probability regime (i.e., elements in $\overline{S}$), which will suffice by a simple reduction from entropy difference to TV (using the lower bound $\min_i p_i, \min_i q_i \geqslant \tilde{\Omega}(\varepsilon/n)$). In more detail, we have

$$\left|\sum_{i\in\overline{S}}(p_i - q_i)\cdot\log\frac{1}{p_i+q_i}\right| \leqslant \sum_{i\in\overline{S}}|p_i - q_i|\cdot\log\frac{1}{p_i+q_i} \leqslant O(\log(n/\varepsilon))\cdot\sum_{i\in\overline{S}}|p_i - q_i|.$$

While this is a task that would in general require a large number of samples, here we can leverage the additional fact that $\overline{S}$ does not contain any heavy elements.

▶ **Lemma 11.** *Let $T \subseteq [n]$ be a set such that $\max_{i\in T}(p_i + q_i) \leqslant C_2 \cdot \frac{\varepsilon}{n^{3/4}\log(n/\varepsilon)}$. Then, testing $p_T = q_T$ versus $\sum_{i\in T}|p_i - q_i| \geqslant \frac{\varepsilon}{\log(n/\varepsilon)}$ (with probability of success $9/10$) can be done with $O\big(n^{3/4}\cdot\log(n/\varepsilon)/\varepsilon\big)$ samples from p, q.*

[8] For simplicity of exposition, we henceforth ignore the factor 4, and consider ε instead of $\varepsilon/4$; a simple rescaling argument gives the original guarantees.

Proof. By [28], we know that $\tilde{O}\left(\frac{n^{3/4}}{\varepsilon}\right)$ samples suffice to distinguish $p_T = q_T$ from $d_{\mathrm{TV}}(p_T, q_T) \geqslant \varepsilon$ with high constant probability. For the sake of completeness, we briefly explain how to adapt their argument to our setting. First, it takes $O(\frac{\log(n/\varepsilon)}{\varepsilon})$ samples to check whether the mass of p on T is larger than $\Omega(\varepsilon/\log(n/\varepsilon))$, and similarly for q: if both have mass smaller that this, we accept, as $\sum_{i\in\overline{S}} |p_i - q_i| \leqslant p(T) + q(T) < \frac{\varepsilon}{\log(n/\varepsilon)}$. If only one of the two puts large enough mass on T, we reject, as this shows that $\sum_{i\in T} |p_i - q_i| \neq 0$. Otherwise, both p, q put enough mass on T, and we continue.

With $O\left(\frac{\log^2(n/\varepsilon)}{\varepsilon^2}\right)$ samples, we then confirm that their probability masses on T approximately match, i.e., that $|p(T) - q(T)| \leqslant O\left(\frac{\varepsilon}{\log(n/\varepsilon)}\right)$. If they do not, we can output reject. Otherwise, we have

$$\begin{aligned}\sum_{i\in T} |p_i - q_i| &= \sum_{i\in T} |p(i|i\in T)p(T) - q(i|i\in T)q(T)| \\ &\leqslant \sum_{i\in T} (|p(i|i\in T)p(T) - p(i|i\in T)q(T)| + |p(i|i\in T)q(T) - q(i|i\in T)q(T)|) \\ &= |p(T) - q(T)| + q(T)\cdot\|p(\cdot|T) - q(\cdot|T)\|_1.\end{aligned}$$

Thus, if $\|p_T - q_T\| \geqslant \Omega\left(\frac{\varepsilon}{\log(n/\varepsilon)}\right)$ and $|p(T) - q(T)| \leqslant O\left(\frac{\varepsilon}{\log(n/\varepsilon)}\right)$ (for suitable choice of constants in the $O(\cdot)$, $\Omega(\cdot)$), then we have that

$$d_{\mathrm{TV}}(p_T, q_T) = \frac{1}{2}\|p(\cdot|T) - q(\cdot|T)\|_1 \geqslant \Omega\left(\frac{\varepsilon}{q(T)\cdot\log(n/\varepsilon)}\right).$$

Recalling that $q(T), p(T) \geqslant \frac{\varepsilon}{\log(n/\varepsilon)}$ and $q(T) = \Theta(p(T))$, we can sample from p_T, q_T via rejection sampling at the cost of an expected $O(1/q(T))$, $O(1/p(T))$ factor in the sample complexity. Thus, we can use an off-the-shelf TV equivalence tester (e.g., from [17, Theorem 1]) to test $d_{\mathrm{TV}}(p(\cdot|T), q(\cdot|T)) \geqslant \Omega\left(\frac{\varepsilon}{q(T)\cdot\log(n/\varepsilon)}\right)$ vs. $p(\cdot|T) = q(\cdot|T)$ at the cost of

$$\max\left\{\frac{n^{2/3}q^{4/3}(T)\log^{4/3}(n/\varepsilon)}{\varepsilon^{4/3}}, \frac{n^{1/2}q^2(T)\log^2(n/\varepsilon)}{\varepsilon^2}\right\}$$

samples from $p(\cdot|T), q(\cdot|T)$; which, by the above discussion, requires an expected

$$\max\left\{\frac{n^{2/3}q^{1/3}(T)\log^{4/3}(n/\varepsilon)}{\varepsilon^{4/3}}, \frac{n^{1/2}q(T)\log^2(n/\varepsilon)}{\varepsilon^2}\right\}$$

samples from p, q (which can be made high-probability via Markov's inequality, paying a constant factor in the sample complexity and a small additional probability of failure). Since $q(T) \leqslant \frac{\varepsilon}{n^{3/4}\log(n/\varepsilon)}\cdot n = \frac{\varepsilon\cdot n^{1/4}}{\log(n/\varepsilon)}$, we finally conclude that taking $O\left(\frac{n^{3/4}\cdot\log(n/\varepsilon)}{\varepsilon}\right)$ samples suffices. ◀

Test 2: $\Lambda_S(p,q) = 0$ vs. $\Lambda_S(p,q) \geqslant \varepsilon$

We now look at the second – and more involved – case, namely, detecting when the entropy difference is far in the high mass regime. That is, we are testing $\Lambda_S(p,q) = 0$ vs. $\Lambda_S(p,q) \geqslant \varepsilon$.

We will define a suitable test statistic, Z, and analyze its expectation (with the goal of bounding its bias and relate it to $\Lambda_S(p,q)$), before bounding its variance to conclude with an application of Chebyshev's inequality. At a high level, our strategy is as follows: either the

empirical entropy difference Z has *low bias*, which means we can simply rely on its output for the test, or it has *high bias*, in which case this can be detected by performing additional checks to detect this bias.

Let $m = \tilde{O}\left(\frac{n^{3/4}}{\varepsilon} + \frac{\log(n/\varepsilon)}{\varepsilon^2}\right)$ be the number of samples taken from p, q, and assume the Poissonized setting: that is, instead of exactly m samples, we take a random $\mathrm{Poi}(m)$ number of samples.[9] Recall that by construction of the set $S \subseteq S_1$, we can assume $p_i + q_i \geqslant \tilde{\Omega}\left(\frac{\varepsilon}{n^{3/4}}\right)$ for all $i \in S$. Focusing on S, we run two further checks, each at cost $O(\log^2 m/\varepsilon^2)$:

1. $|p(S) - q(S)| \leqslant O(\varepsilon/\log m)$, which implies $|\sum_{i\in S}(p_i - q_i)| \log m \leqslant O(\varepsilon)$.
2. $\|p - q\|_2^2 \leqslant O(\varepsilon^2/\log^2 m)$, which is checked via testing at distance $\varepsilon/\log m$ using an ℓ_2-closeness tester (e.g., via learning p, q in ℓ_2 distance, see for instance [16]).

If either check rejects, we reject; otherwise we proceed assuming both guarantees hold.

Letting X_i (resp. Y_i) be the number of occurrences of element $i \in S$ among the samples from p (resp. from q), we define our entropy difference estimator to be

$$Z := \sum_{i\in S} \frac{X_i - Y_i}{m} \log \frac{1}{X_i + Y_i},$$

and we write $Z_i := \frac{X_i - Y_i}{m} \log \frac{1}{X_i+Y_i}$ for the i-th summand and let $Z_i = 0$ when $X_i + Y_i = 0$. By the above Poissonization procedure, we have $X_i \sim \mathrm{Poi}(m \cdot p_i)$ and $Y_i \sim \mathrm{Poi}(m \cdot q_i)$ for all i, and the $(X_i)_i$ (resp. $(Y_i)_i$) are independent. We first bound the bias of this estimator Z:

▶ **Lemma 12.** *For any set $S \subseteq [n]$,*

$$\left|\sum_{i\in S}(p_i - q_i)\log\tfrac{1}{m(p_i+q_i)} - \mathbb{E}[Z]\right| \leqslant \sum_{i\in S}\frac{|p_i - q_i|}{m(p_i + q_i)}.$$

Proof. Let $J_i := X_i + Y_i \sim \mathrm{Poi}(\lambda_i)$ with $\lambda_i := m(p_i + q_i)$. Conditional on $J_i = j$, the count X_i is $\mathrm{Bin}(j, p_i/(p_i + q_i))$-distributed, so $\mathbb{E}[X_i - Y_i \mid J_i] = J_i \cdot (p_i - q_i)/(p_i + q_i)$. By the law of total expectation, pulling the factor $\log(1/J_i)$ outside the inner expectation,

$$\mathbb{E}[Z_i] = \tfrac{1}{m}\,\mathbb{E}\left[(X_i - Y_i)\log\tfrac{1}{J_i}\right] = \tfrac{1}{m}\,\mathbb{E}\left[\log\tfrac{1}{J_i}\cdot\mathbb{E}[X_i - Y_i \mid J_i]\right] = \tfrac{p_i - q_i}{m(p_i+q_i)}\,\mathbb{E}[J_i\log(1/J_i)].$$

We then apply the Poisson identity (Fact 5) $\mathbb{E}[J_i f(J_i)] = \lambda_i\,\mathbb{E}[f(J_i + 1)]$, taking $f(j) = \log(1/j)$:

$$\mathbb{E}[Z_i] = \tfrac{p_i - q_i}{m(p_i+q_i)} \cdot \lambda_i\,\mathbb{E}[\log(1/(J_i + 1))] = -(p_i - q_i)\,\mathbb{E}[\log(J_i + 1)].$$

Subtracting the target $(p_i - q_i)\log(1/\lambda_i)$,

$$(p_i - q_i)\log\tfrac{1}{\lambda_i} - \mathbb{E}[Z_i] = (p_i - q_i)\,\mathbb{E}\left[\log\tfrac{J_i+1}{\lambda_i}\right].$$

We proceed to bound $\mathbb{E}[\log((J_i + 1)/\lambda_i)]$ from above and below. By Jensen's inequality applied to the concave function log,

$$\mathbb{E}\left[\log\tfrac{J_i+1}{\lambda_i}\right] \leqslant \log\left(1 + \tfrac{1}{\lambda_i}\right) \leqslant \tfrac{1}{\lambda_i}.$$

[9] This standard technique is valid as it only affects the sample complexity by constant terms and an arbitrarily small error probability adjustment, since a Poisson random variable tightly concentrates around its mean. Its advantage is to provide additional independence between the empirical counts of each element, which is useful in the analysis.

For the lower bound, the inequality $\log x \geqslant 1 - 1/x$ applied to $x = (J_i + 1)/\lambda_i$, together with $\mathbb{E}[1/(J_i + 1)] = (1 - e^{-\lambda_i})/\lambda_i$ for $J_i \sim \mathrm{Poi}(\lambda_i)$, gives

$$\mathbb{E}\Big[\log \tfrac{J_i+1}{\lambda_i}\Big] \geqslant 1 - \lambda_i \,\mathbb{E}\Big[\tfrac{1}{J_i+1}\Big] = e^{-\lambda_i} \geqslant 0.$$

Combining these findings gives $|\mathbb{E}[\log((J_i+1)/\lambda_i)]| \leqslant 1/\lambda_i$ (using $e^{-\lambda_i} \leqslant 1/\lambda_i$ for all $\lambda_i > 0$), so

$$\big|(p_i - q_i)\log \tfrac{1}{\lambda_i} - \mathbb{E}[Z_i]\big| \leqslant \tfrac{|p_i - q_i|}{\lambda_i}.$$

Summing over $i \in S$ yields the claim. ◀

The bias of our estimator is determined by the RHS of Lemma 12; note that if $p = q$, the RHS simplifies to 0. As a result, a natural idea would be to use this RHS as a proxy, and to check if the RHS is noticeably large using another test: if it is larger than $\geqslant \Omega(\varepsilon)$, we can immediately reject, as this is clear evidence that $p \neq q$. We do so in the following lemma:

▶ **Lemma 13.** *Suppose that $d_H^2(p, q) \leqslant O(\varepsilon)$. Then, with $s = O(n^{3/4}\log(n/\varepsilon)/\varepsilon)$ samples from p, q, one can distinguish between the following two cases with high constant probability:*

1. *$p = q$ (which implies $\sum_{i\in S} \frac{|p_i - q_i|}{m(p_i+q_i)} = 0$).*
2. *$\sum_{i\in S} \frac{|p_i - q_i|}{(p_i+q_i)} \geqslant \Omega(m\varepsilon)$.*

Proof. We use the test statistic

$$T := \sum_{i\in S} \frac{(X_i - Y_i)^2 - (X_i + Y_i)}{X_i + Y_i},$$

where $X_i \sim \mathrm{Poi}(s \cdot p_i)$ and $Y_i \sim \mathrm{Poi}(s \cdot q_i)$. We show:

- In the first case, $\mathbb{E}[T] = 0$, while in the second case $\mathbb{E}[T] \geqslant \Omega(\sqrt{n})$.
- $\mathrm{Var}[T] \leqslant O(n)$ in both cases.

Chebyshev's inequality then will enable us to separate the two cases by thresholding T at a value scaling as $\Theta(\sqrt{n})$.

Analyzing $\mathbb{E}[T]$. The first part is similar to the analysis from [17, Lemma 4], which relies on the same statistic T (for a different purpose): write $\delta_i := (p_i - q_i)/(p_i + q_i)$ and $J_i := X_i + Y_i$. Conditional on $J_i = j$, we have $Y_i = j - X_i$ so $X_i - Y_i = 2X_i - j$, and $X_i \sim \mathrm{Bin}(j, p_i/(p_i+q_i))$. Hence,

$$\mathbb{E}[X_i - Y_i \mid J_i = j] = j\delta_i,$$

and the binomial variance $\mathrm{Var}[X_i \mid J_i = j] = j \cdot \frac{p_i q_i}{(p_i+q_i)^2}$ gives

$$\mathrm{Var}[X_i - Y_i \mid J_i = j] = 4j \cdot \tfrac{p_i q_i}{(p_i+q_i)^2} = j(1 - \delta_i^2).$$

Combining, $\mathbb{E}[(X_i - Y_i)^2 \mid J_i = j] = \mathrm{Var}[X_i - Y_i \mid J_i = j] + \mathbb{E}[X_i - Y_i \mid J_i = j]^2 = j(1 - \delta_i^2) + j^2\delta_i^2$. The i-th summand of T is $\frac{(X_i - Y_i)^2 - J_i}{J_i}$, which we set to 0 when $J_i = 0$ (the 0/0 case). For $J_i = j \geqslant 1$,

$$\mathbb{E}\left[\frac{(X_i - Y_i)^2 - J_i}{J_i} \,\middle|\, J_i = j\right] = \frac{(j^2 - j)\delta_i^2}{j} = (j-1)\delta_i^2.$$

Taking the outer expectation, with the indicator $\mathbf{1}\{J_i \geqslant 1\}$ enforcing the 0/0 convention,

$$\mathbb{E}[T] = \sum_{i\in S} \delta_i^2 \cdot \mathbb{E}[(J_i - 1)\mathbf{1}\{J_i \geqslant 1\}] = \sum_{i\in S} \delta_i^2 \big(\lambda_i - 1 + e^{-\lambda_i}\big),$$

where $\mathbb{E}[(J_i - 1)\mathbf{1}\{J_i \geqslant 1\}] = \mathbb{E}[J_i] - \Pr[J_i \geqslant 1] = \lambda_i - (1 - e^{-\lambda_i})$.

In particular, in the first case (when $p = q$), $\delta_i = 0$ for every $i \in S$, and so $\mathbb{E}[T] = 0$. We therefore now focus on the second case, for which we depart from [17]. By the construction of S (Lemma 10), $p_i + q_i \geqslant \Omega(\frac{\varepsilon}{n^{3/4}\log(n/\varepsilon)})$ for every $i \in S$, so choosing $s = \Omega(n^{3/4} \cdot \log(n/\varepsilon)/\varepsilon)$ with a sufficiently large constant gives $\lambda_i = s(p_i + q_i) \geqslant 2$ for all $i \in S$. The bound $\lambda - 1 + e^{-\lambda} \geqslant \lambda - 1 \geqslant \lambda/2$ for $\lambda \geqslant 2$ then yields

$$\mathbb{E}[T] \geqslant \tfrac{1}{2} \sum_{i\in S} \lambda_i \delta_i^2 = \tfrac{s}{2} \sum_{i\in S} \frac{(p_i - q_i)^2}{p_i + q_i}.$$

By Cauchy–Schwarz,

$$\sum_{i\in S} \frac{(p_i - q_i)^2}{p_i + q_i} \geqslant \frac{1}{n} \left(\sum_{i\in S} \frac{|p_i - q_i|}{\sqrt{p_i + q_i}} \right)^2.$$

Since $m(p_i + q_i) \geqslant 1$ (in view of $i \in S$ and $m = \tilde{\Omega}(\frac{n^{3/4}}{\varepsilon})$), we have $\frac{1}{\sqrt{p_i+q_i}} \geqslant \frac{1}{\sqrt{m}(p_i+q_i)}$. Applying this termwise and using the definition of the second case,

$$\sum_{i\in S} \frac{|p_i - q_i|}{\sqrt{p_i + q_i}} \geqslant \frac{1}{\sqrt{m}} \sum_{i\in S} \frac{|p_i - q_i|}{p_i + q_i} \geqslant \Omega(\sqrt{m}\varepsilon).$$

Combining,

$$\mathbb{E}[T] \geqslant \frac{s}{2n} \cdot \Omega(m\varepsilon^2) = \Omega\left(\frac{sm\varepsilon^2}{n}\right),$$

which is $\Omega(\sqrt{n} + \frac{\log^2(n/\varepsilon)}{n^{1/4}\varepsilon}) \geqslant \Omega(\sqrt{n})$ given the settings of $s = \tilde{\Theta}(n^{3/4}/\varepsilon)$ and $m = \tilde{\Theta}(n^{3/4}/\varepsilon + \frac{\log^2(n/\varepsilon)}{\varepsilon^2})$.

Upper bound on $\mathbf{Var}[T]$. The variance bound from [17, Lemma 5] gives

$$\mathrm{Var}[T] \leqslant 2\min\{n, s\} + 5s \sum_{i\in S} \frac{(p_i - q_i)^2}{p_i + q_i}.$$

Under the lemma's assumption on the Hellinger distance, $\sum_{i\in S}(p_i - q_i)^2/(p_i + q_i) = \Theta(d_H^2(p, q)) \leqslant O(\varepsilon)$, so the second term is $\widetilde{O}(s \cdot \varepsilon)$, and $\mathrm{Var}[T] \leqslant O(n + s \cdot \varepsilon)$, where $s = \tilde{O}\left(n^{3/4}/\varepsilon\right)$. This scaling of s leads to the simplified bound $\mathrm{Var}[T] \leqslant O(n)$.

Concluding via Chebyshev. In the first case, $\mathbb{E}[T] = 0$ and $\mathrm{Var}[T] \leqslant O(n)$, so $|T| \leqslant O(\sqrt{n})$ with probability $\geqslant 9/10$. In the second case, $\mathbb{E}[T] \geqslant C\sqrt{n}$ and $\mathrm{Var}[T] \leqslant O(n) \leqslant \mathbb{E}[T]^2/C'$, so $T \geqslant C''\sqrt{n}$ with probability $\geqslant 9/10$. Thresholding at $\Theta(\sqrt{n})$ thus distinguishes the two cases. ◀

Having used the RHS of Lemma 12 as a proxy for the bias, and checked that it is small (and rejected otherwise), we are not yet done: we could still have both small bias for Z (so that $\mathbb{E}[Z] \approx \sum_{i\in S}(p_i - q_i)\log\frac{1}{m(p_i+q_i)}$) *and* large $\Lambda_S(p, q) = \left|\sum_{i\in S}(p_i - q_i)\log\frac{1}{m(p_i+q_i)}\right|$. However, now, since we know that $|\mathbb{E}[Z]| \approx \Lambda_S(p, q)$, to conclude it suffices to run one last check: namely, that Z is small enough.

Indeed, we know now that Z has small bias: if we can also show, via a variance argument, that Z concentrates around its expectation, then we will have $|Z| \approx_{\text{w.h.p.}} |\mathbb{E}[Z]| \approx \Lambda_S(p,q)$, and checking that $|Z|$ is small will ensure (with high probability) that we can guarantee that $\Lambda_S(p,q)$ is small. Thus, our last missing piece is an upper bound on $\mathrm{Var}[Z]$, which will let us show that Z concentrates around its expectation.

▶ **Lemma 14.** *The variance of the entropy difference estimator Z satisfies*

$$\mathrm{Var}[Z] \leqslant O(\log^2 m)\|p-q\|_2^2 + \frac{\log^2 m}{m} \leqslant O(\varepsilon^2).$$

Proof. Under Poissonization, the counts $\{X_i\}_{i\in S}$ and $\{Y_i\}_{i\in S}$ are mutually independent across coordinates. As a result, the Z_i's are also independent, and

$$\mathrm{Var}[Z] = \sum_{i\in S} \mathrm{Var}[Z_i] \leqslant \sum_{i\in S} \mathbb{E}[Z_i^2].$$

Fix any $i \in S$, letting $J_i := X_i + Y_i \sim \mathrm{Poi}(m(p_i+q_i))$, and defining the event $\mathcal{E}_i := \{J_i \leqslant 3m\}$. Since $m(p_i+q_i) \leqslant m$, the Chernoff bound for Poisson random variables gives $\Pr[\mathcal{E}_i^c] \leqslant e^{-\Omega(m)}$. Then:

- On $\mathcal{E}_i$: $|\log(1/J_i)| \leqslant \log(3m)$ for $J_i \neq 0$, and noting from before that $Z_i = 0$ when $J_i = 0$, we have $|Z_i|\,\mathbf{1}_{\mathcal{E}_i} \leqslant \frac{\log(3m)}{m}|X_i - Y_i|$. Squaring and taking expectations,

  $$\mathbb{E}[Z_i^2\,\mathbf{1}_{\mathcal{E}_i}] \leqslant \tfrac{\log^2(3m)}{m^2}\,\mathbb{E}[(X_i-Y_i)^2] = \tfrac{\log^2(3m)}{m^2}\big(m(p_i+q_i) + m^2(p_i-q_i)^2\big),$$

  using $\mathrm{Var}[X_i - Y_i] = m(p_i+q_i)$ and $\mathbb{E}[X_i - Y_i] = m(p_i - q_i)$ (moments of Poisson r.v.'s).
- On $\mathcal{E}_i^c$: from $|X_i - Y_i| \leqslant J_i$ and $|\log(1/J_i)| = \log J_i \leqslant J_i$ (valid since $J_i > 3m \geqslant 1$), we get $|Z_i|\,\mathbf{1}_{\mathcal{E}_i^c} \leqslant J_i^2/m$. Combining Cauchy–Schwarz with the moment bound $\mathbb{E}[J_i^8] = O(m^8)$ (since Poisson moments are polynomials in the mean, by Fact 5) and the tail bound $\Pr[\mathcal{E}_i^c] \leqslant e^{-\Omega(m)}$,

  $$\mathbb{E}[Z_i^2\,\mathbf{1}_{\mathcal{E}_i^c}] \leqslant \tfrac{1}{m^2}\,\mathbb{E}[J_i^4\,\mathbf{1}_{\mathcal{E}_i^c}] \leqslant \tfrac{1}{m^2}\sqrt{\mathbb{E}[J_i^8]\,\Pr[\mathcal{E}_i^c]} \leqslant e^{-\Omega(m)}.$$

  Summing over $i \in S$, using $\log^2(3m) \leqslant 4\log^2 m$ (for $m \geqslant 3$) and $\sum_i (p_i+q_i) \leqslant 2$,

  $$\mathrm{Var}[Z] \leqslant O(\log^2 m)\,\|p-q\|_2^2 + \tfrac{O(\log^2 m)}{m} + n\cdot e^{-\Omega(m)}.$$

The last term is absorbed into $O(\frac{\log^2 m}{m})$ since $m \geqslant \log n$ makes $n\cdot e^{-\Omega(m)} \leqslant 1/m$. ◀

Combining Lemma 12 and Lemma 13, and conditioning on all tests passing, the expectation of Z has small bias:

$$\left|\mathbb{E}[Z] - \sum_i (p_i - q_i)\log\tfrac{1}{(p_i+q_i)}\right| \leqslant O(\varepsilon).$$

By Lemma 14, $\mathrm{Var}[Z] \leqslant O(\varepsilon^2)$, so Chebyshev gives $|Z - \mathbb{E}[Z]| \leqslant O(\varepsilon)$ with probability at least $9/10$. Combining via the triangle inequality,

$$\Pr\left[\left|Z - \sum_i (p_i-q_i)\log\tfrac{1}{(p_i+q_i)}\right| \leqslant O(\varepsilon)\right] \geqslant \tfrac{9}{10}.$$

To conclude this subcase. We choose a threshold for $|Z|$ at a suitable constant multiple of ε that distinguishes the two cases. If $\Lambda(p,q) = \left|\sum_{i\in S}(p_i - q_i)\log\frac{1}{(p_i+q_i)}\right| = 0$, then with probability at least 9/10 the test on T from Lemma 13 will not reject and $|Z| \leqslant O(\varepsilon)$, and so this overall test accepts. If $d_H^2(p,q) \leqslant O(\varepsilon)$ and $\left|\sum_{i\in S}(p_i - q_i)\log\frac{1}{(p_i+q_i)}\right| \geqslant \Omega(\varepsilon)$, then with probability at least 9/10 either the test on T from Lemma 13 rejects, or it does not and then $|Z| \geqslant \Omega(\varepsilon)$, and this overall test rejects.

Wrapping up

Putting the above ingredients together:

- If $p = q$, then with high constant probability, (1) the Hellinger test on $d_H(p,q)$ accepts, and (2) (a) the set S is correctly identified, (b) the test on $\Lambda_{\overline{S}}(p,q)$ accepts, (c) (i) the test on T (checking for low bias for Z) accepts, and (ii) the test on Z (test on $\Lambda_S(p,q)$ assuming low bias for Z) accepts. Overall, the algorithm accepts with probability at least 9/10 by a union bound.
- If $|H(p) - H(q)| \geqslant \varepsilon$, then with high constant probability (1) either the Hellinger test on $d_H(p,q)$ rejects; or it accepts because $d_H(p,q)$ is small, in which case $\Lambda(p,q)$ must be large; then, with high constant probability (2) (a) the set S is correctly identified, (b) the test on $\Lambda_{\overline{S}}(p,q)$ rejects, or it accepts because $\Lambda_{\overline{S}}(p,q)$ is small, which means $\Lambda_S(p,q)$ must be large: in which case (c) (i) the test on T (checking for low bias for Z assuming $d_H(p,q)$ is small) rejects, or it accepts because the bias is small, in which case (ii) the test on Z (test on $\Lambda_S(p,q)$ assuming low bias for Z) rejects. Overall, the algorithm rejects with probability at least 9/10 by a union bound.

Overall, each subtest in Algorithm 1 uses no more than $O\left(\frac{n^{3/4}\log(n/\varepsilon)\log n}{\varepsilon}\right)$ or $O\left(\frac{\log^2(n/\varepsilon)}{\varepsilon^2}\right)$ samples, and hence the total sample complexity of our algorithm is

$$O\left(\frac{n^{3/4}\log(n/\varepsilon)\log n}{\varepsilon} + \frac{\log^2(n/\varepsilon)}{\varepsilon^2}\right).$$

All our test statistics can be computed in $O(n)$ time once all X_i, Y_i are collected.[10] To compute X_i, Y_i, one can first sort over all samples which takes $O(s\log s)$ time, where s is the number of samples in one of the subtests, and then sum them up. Thus, the time complexity of our algorithm is $O(N\log N)$, where N is the total number of samples. This concludes the proof of Theorem 8.

3 Bayesian Network Equivalence Testing

In this section, we show how to leverage our Entropy Equivalence Testing (EET) algorithm (Algorithm 2) to obtain an *equivalence testing* (closeness testing) algorithm for bounded-degree Bayesian networks (Bayes nets). Specifically, the problem we consider is the following:

[10] The statistics for testing Hellinger square is the same as T (albeit with different threshold; see [24, Theorem 5]). T can be computed in $O(n)$ time once all X_i, Y_i are known. Computing $S \leftarrow \left\{ i \in [n] : X_i + Y_i \geqslant \Omega(m_1 \cdot \varepsilon/(n^{3/4}\log(n/\varepsilon))) \right\}$ can be done in $O(n)$ time and calculating the empirical of S can be implemented in $O(m_2)$ time. As for the conditional TV tester, the rejection sampling will cost as many samples as we take in total, and for TV closeness testing we use the same statistic T (see, e.g., [24, Theorem 4]). For ℓ_2 tester, the statistic is also computable in $O(n)$ time [17, Section 3]. Assuming that we can compute $\log(x)$ in constant time, one can compute the statistic Z in $O(n)$ time.

Bayes-Net Equivalence Testing: Given sample access to distributions p, q over $\{0,1\}^n$, each Markov with respect to a DAG of in-degree at most d (the DAG is unknown, and possibly distinct between p and q), and parameters $\varepsilon \in (0, 1/2]$ and $\delta \in (0, 1]$, distinguish, with probability at least $1 - \delta$, between the following two cases: (1) $p = q$, and (2) $d_{\mathrm{TV}}(p, q) \geq \varepsilon$.

▶ **Definition 15.** *A* projection of a Bayes net p *on* $\{0,1\}^n$ *onto a DAG* G *is denoted by* p_G*, and is defined by its probability mass function (PMF) as follows:*

$$p_G(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i \mid \Pi_i^G),$$

where Π_i^G *is the set of parents of* X_i *in* G*. Moreover, with a slight abuse of notation, we write* p_{X_i,Π_i} *or* $p_{X_i,\Pi_i}(x_i, \pi_i)$ *for the marginal distribution of* p *on the subset* $\{X_i, \Pi_i\}$*.*

The baseline here is the testing-by-learning approach, whereby one learns the two unknown in-degree d Bayes nets to TV distance $\frac{\varepsilon}{3}$ and checks the distance between the two learned hypotheses. It is known that learning an in-degree d Bayes net in total variation distance can be done in $\tilde{O}(2^d n/\varepsilon^2)$ samples (see [12, Appendix A]): to the best of our knowledge, the time complexity of any learning algorithm achieving this bound is exponential, namely, $2^{\tilde{\Omega}(n)}$. In contrast, we obtain a testing algorithm with drastically better running time (for all $d = o(n)$), and significantly better sample complexity whenever $d \gg \log(n/\varepsilon^2)$.

▶ **Theorem 16** (Theorem 4, restated)**.** *There exists an algorithm (Algorithm 2) that, given sample access to two Bayesian networks* p *and* q *over* $\{0,1\}^n$ *with in-degree at most* d*, as well as a parameter* $\varepsilon \in (0, 1]$*, takes*

$$s = \tilde{O}\left(\min\left\{\frac{2^{3d/4}n}{\varepsilon^2}, \frac{2^{2d/3}n^{4/3}}{\varepsilon^{8/3}}\right\} + \frac{n^2}{\varepsilon^4}\right),$$

samples from p, q*, and distinguishes with probability at least* $2/3$ *between (1)* $p = q$ *and (2)* $d_{\mathrm{TV}}(p, q) \geqslant \varepsilon$*. Moreover, the algorithm runs in time* $\tilde{O}(n^{d+1}s)$*.*

Proof. The pseudocode is provided in Algorithm 2. Due to space constraints, we defer the proof of correctness to the full version [14]. ◀

▶ Remark 17. We have an interesting dichotomy: our algorithm is efficient (polynomial in n) only if d is constant. However, it is worse than its learning counterpart when d is constant: n^2 (our test) vs. n (learning). On the other hand, if $d = \Omega(\log n)$, then our tester will be more efficient than learning sample-wise, but its runtime becomes $n^{\Omega(\log n)}$, and is thus quasi-polynomial.

As identity testing is a special case of closeness testing, our above analysis can straightforwardly be ported to yield an identity testing algorithm (using an algorithm for identity testing in Hellinger instead of a closeness one as the first building block). An immediate consequence of this is to yield an improved identity tester for Bayes net with sample complexity $\tilde{O}(\frac{2^{d/2}n}{\varepsilon^2} + \frac{n^2}{\varepsilon^4})$, thus improving on the prior bound $\tilde{O}(\frac{2^{d/2}n^{3/2}}{\varepsilon^2} + \frac{n^2}{\varepsilon^4})$ [15, Theorem 1.2].

Algorithm 2 Closeness testing for Bayesian networks.

Require: Sample access to p and q, both in-degree d Bayesian networks over $\{0,1\}^n$; accuracy parameter $\varepsilon \in (0,1]$.

1: Set $\varepsilon_1 \leftarrow \Theta\left(\frac{\varepsilon^2}{n}\right)$, $\varepsilon_2 \leftarrow \Theta\left(\frac{\varepsilon^2}{d\cdot n\cdot\log(dn/\varepsilon)}\right)$, $\delta \leftarrow \frac{1}{20\binom{n}{d+1}}$

2: Set

$$m \leftarrow \tilde{O}\left(\min\left\{\frac{2^{3d/4}\cdot n}{\varepsilon^2}, \frac{2^{2d/3}\cdot n^{4/3}}{\varepsilon^{8/3}}\right\}\right) + O\left(\frac{d^3n^2\cdot\log^2(n/\varepsilon)}{\varepsilon^4}\right)$$

3: Take a multiset set S_p (resp. S_q) of m samples from $\tilde{p}$ (resp. $\tilde{q}$), defined by

$$\tilde{p} := \left(1 - \frac{C_3\cdot\varepsilon^2}{dn\log(n/\varepsilon)}\right)p + \frac{C_3\cdot\varepsilon^2}{dn\log(n/\varepsilon)}\cdot U \quad \text{and} \quad \tilde{q} := \left(1 - \frac{C_3\cdot\varepsilon^2}{dn\log(n/\varepsilon)}\right)q + \frac{C_3\cdot\varepsilon^2}{dn\log(n/\varepsilon)}\cdot U$$

4: **for all** subset $T \subseteq [n]$ with $|T| = d+1$ **do** ▷ $\binom{n}{d+1}$ possible subsets

5: Let $\mathcal{T} \leftarrow \{0,1\}^T$

6: Test $\tilde{p}_{\mathcal{T}} = \tilde{q}_{\mathcal{T}}$ vs. $|H(\tilde{p}_{\mathcal{T}}) - H(\tilde{q}_{\mathcal{T}})| \geqslant \varepsilon_1$ with failure probability δ, using S_p, S_q

7: ▷ By Corollary 2

8: Test $\tilde{p}_{\mathcal{T}} = \tilde{q}_{\mathcal{T}}$ vs. $d_H^2(\tilde{p}_{\mathcal{T}}, \tilde{q}_{\mathcal{T}}) \geqslant \varepsilon_2$ with failure probability δ, using S_p, S_q ▷ By [28]

9: **if** either test rejects **then return** reject

10: **return** accept

▶ **Corollary 18.** *Given sample access to a Bayesian networks p and a full description of Bayesian network q over $\{0,1\}^n$ with in-degree at most d, as well as a parameter $\varepsilon \in (0,1]$, there is an algorithm that takes*

$$s = \tilde{O}\left(\frac{2^{d/2}n}{\varepsilon^2} + \frac{n^2}{\varepsilon^4}\right),$$

samples from p, and distinguishes with probability at least $2/3$ between (1) $p = q$ and (2) $d_{\mathrm{TV}}(p,q) \geqslant \varepsilon$. Moreover, the algorithm runs in time $\tilde{O}(n^{d+1}s)$.

4 Lower Bound

In this section, we establish our sample complexity lower bound for EET (Theorem 3). The $\widetilde{\Omega}(\min\{n^{3/4}/\varepsilon, n^{2/3}/\varepsilon^{4/3}\})$ term follows from a reduction to mutual information (MI) testing combined with [37, Corollary 6.2], and the $\Omega(\log^2 n/\varepsilon^2)$ term follows directly from [15, Lemma 2.7], whose two-point construction is also a hard instance for EET.

For a joint distribution P_{AC} over $[k_A]\times[k_C]$ with marginals P_A, P_C, the *mutual information* between coordinates is

$$I(A:C) := H(A) + H(C) - H(A,C),$$

and equals zero if and only if $P_{AC} = P_A \otimes P_C$. The MI testing problem asks, given sample access to P_{AC}, to distinguish $I(A:C) = 0$ from $I(A:C) \geqslant \varepsilon$ with probability at least $2/3$. [37, Corollary 6.2] established the lower bound

$$\widetilde{\Omega}\left(\min\left\{\frac{k_A^{3/4}k_C^{1/4}}{\varepsilon}, \frac{k_A^{2/3}k_C^{1/3}}{\varepsilon^{4/3}}\right\}\right)$$

for this task.

Reduction from MI testing to EET

▶ **Lemma 19.** *Let $k_A, k_C \geqslant 2$ and $n = k_A k_C$. Any* EET *algorithm $\mathcal{A}$ over $[n]$ with sample complexity T yields an* MI *testing algorithm over $[k_A] \times [k_C]$ with sample complexity $3T$ and the same success probability.*

Proof. Map $[k_A] \times [k_C]$ with $[n]$ via some fixed bijection. Using $\mathcal{A}$, we construct a MI tester for P_{AC} over $[k_A] \times [k_C]$ as follows:

1. Draw $3T$ i.i.d. samples $(a_1, c_1), \ldots, (a_{3T}, c_{3T})$ from P_{AC}.
2. Form T samples of $P := P_{AC}$, denoted by (a_i, c_i) for $i = 1, \ldots, T$.
3. Form T samples of $Q := P_A \otimes P_C$, denoted by (a_{T+2i-1}, c_{T+2i}) for $i = 1, \ldots, T$. Since the samples are i.i.d., the a- and c-coordinates of each pair are independent draws from P_A and P_C respectively.
4. Run $\mathcal{A}$ on the T samples of P and the T samples of Q.

Observe that $|H(P) - H(Q)| = |H(A, C) - H(A) - H(C)| = I(A : C)$. Then:

- If $I(A : C) = 0$, then $P_{AC} = P_A \otimes P_C$, so $P = Q$ and $\mathcal{A}$ returns accept.
- If $I(A : C) \geqslant \varepsilon$, then $|H(P) - H(Q)| \geqslant \varepsilon$ and $\mathcal{A}$ returns reject. ◀

▶ **Theorem 20** (EET lower bound: Theorem 3, restated)**.** *Any algorithm that, given sample access to two unknown distributions p, q over $[n]$ and parameter $\varepsilon \in (0, 1]$, distinguishes $p = q$ from $|H(p) - H(q)| \geqslant \varepsilon$ with probability at least $2/3$ requires*

$$\widetilde{\Omega}\left(\min\left\{\frac{n^{3/4}}{\varepsilon}, \frac{n^{2/3}}{\varepsilon^{4/3}}\right\}\right) + \Omega\left(\frac{\log^2 n}{\varepsilon^2}\right)$$

samples from each of p and q.

Proof. Apply Lemma 19 with $k_A = n/2$ and $k_C = 2$. Combining with [37, Corollary 6.2] gives the first term:

$$\widetilde{\Omega}\left(\min\left\{\frac{(n/2)^{3/4} \cdot 2^{1/4}}{\varepsilon}, \frac{(n/2)^{2/3} \cdot 2^{1/3}}{\varepsilon^{4/3}}\right\}\right) = \widetilde{\Omega}\left(\min\left\{\frac{n^{3/4}}{\varepsilon}, \frac{n^{2/3}}{\varepsilon^{4/3}}\right\}\right).$$

For the second term, [15, Lemma 2.7] exhibits distributions p, q over $[n]$ with $|H(p) - H(q)| \geqslant \varepsilon$ that no algorithm can distinguish from $p = q$ using fewer than $\Omega(\log^2 n/\varepsilon^2)$ samples from p when q is given. Any EET algorithm using T samples from each of p, q yields such a tester by simulating T samples from the known q, so EET also requires $\Omega(\log^2 n/\varepsilon^2)$ samples. ◀

References

1 Jayadev Acharya, Constantinos Daskalakis, and Gautam Kamath. Optimal testing for properties of distributions. In *NeurIPS*, pages 3591–3599, 2015. URL: https://proceedings.neurips.cc/paper/2015/hash/1f36c15d6a3d18d52e8d493bc8187cb9-Abstract.html.

2 Jayadev Acharya, Alon Orlitsky, Ananda Theertha Suresh, and Himanshu Tyagi. The complexity of estimating rényi entropy. In *SODA*, pages 1855–1869. SIAM, 2015. doi:10.1137/1.9781611973730.124.

3 Tugkan Batu, Sanjoy Dasgupta, Ravi Kumar, and Ronitt Rubinfeld. The complexity of approximating the entropy. *SIAM J. Comput.*, 35(1):132–150, 2005. doi:10.1137/S0097539702403645.

4 Tugkan Batu, Lance Fortnow, Eldar Fischer, Ravi Kumar, Ronitt Rubinfeld, and Patrick White. Testing random variables for independence and identity. In *FOCS*, pages 442–451. IEEE Computer Society, 2001. doi:10.1109/SFCS.2001.959920.

5 Tugkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D. Smith, and Patrick White. Testing that distributions are close. In *FOCS*, pages 259–269. IEEE Computer Society, 2000. doi:10.1109/SFCS.2000.892113.

6 Arnab Bhattacharyya, Clément L. Canonne, and Joy Qiping Yang. Independence testing for bounded degree bayesian networks. In *NeurIPS*, 2022.

7 Arnab Bhattacharyya, Davin Choo, Sutanu Gayen, and Dimitrios Myrisiotis. Learnability of parameter-bounded bayes nets. In *AAAI*, pages 15559–15566. AAAI Press, 2025. doi:10.1609/AAAI.V39I15.33708.

8 Arnab Bhattacharyya, Sutanu Gayen, Philips George John, Sayantan Sen, and N. V. Vinodchandran. Distribution learning meets graph structure sampling. In *Advances in Neural Information Processing Systems*, volume 38, pages 21411–21432, 2025.

9 Arnab Bhattacharyya, Sutanu Gayen, Eric Price, Vincent Y. F. Tan, and N. V. Vinodchandran. Near-optimal learning of tree-structured distributions by chow and liu. *SIAM J. Comput.*, 52(3):761–793, 2023. doi:10.1137/22M1489678.

10 Clément L. Canonne. *A Survey on Distribution Testing: Your Data is Big. But is it Blue?* Number 9 in Graduate Surveys. Theory of Computing Library, 2020. doi:10.4086/toc.gs.2020.009.

11 Clément L. Canonne. Topics and techniques in distribution testing: A biased but representative sample. *Found. Trends Commun. Inf. Theory*, 19(6):1032–1198, 2022. doi:10.1561/0100000114.

12 Clément L Canonne, Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. Testing Bayesian networks. In *Conference on Learning Theory*, pages 370–448. PMLR, 2017. URL: http://proceedings.mlr.press/v65/canonne17a.html.

13 Clément L. Canonne, Ilias Diakonikolas, Daniel M. Kane, and Alistair Stewart. Testing conditional independence of discrete distributions. In *STOC*, pages 735–748. ACM, 2018. doi:10.1145/3188745.3188756.

14 Clément L. Canonne, Yash Pote, Jonathan Scarlett, and Joy Qiping Yang. Entropy equivalence testing. *CoRR*, abs/2605.23225, 2026. doi:10.48550/arXiv.2605.23225.

15 Clément L Canonne and Joy Q Yang. Entropy testing and its application to testing Bayesian networks. *Advances in Neural Information Processing Systems*, 2024.

16 Clément L. Canonne. A short note on learning discrete distributions, 2020. arXiv:2002.11457.

17 Siu On Chan, Ilias Diakonikolas, Paul Valiant, and Gregory Valiant. Optimal algorithms for testing closeness of discrete distributions. In *SODA*, pages 1193–1203. SIAM, 2014. doi:10.1137/1.9781611973402.88.

18 David Maxwell Chickering, David Heckerman, and Christopher Meek. Large-sample learning of bayesian networks is np-hard. *J. Mach. Learn. Res.*, 5:1287–1330, 2004. URL: https://jmlr.org/papers/volume5/chickering04a/chickering04a.pdf.

19 Davin Choo, Joy Qiping Yang, Arnab Bhattacharyya, and Clément L. Canonne. Learning bounded-degree polytrees with known skeleton. In *ALT*, Proceedings of Machine Learning Research, pages 402–443. PMLR, 2024. URL: https://proceedings.mlr.press/v237/choo24a.html.

20 C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. Inf. Theory*, 14(3):462–467, 1968. doi:10.1109/TIT.1968.1054142.

21 Imre Csiszár and János Körner. *Information Theory: Coding Theorems for Discrete Memoryless Systems*. Cambridge University Press, 2011.

22 Sanjoy Dasgupta. Learning polytrees. In *UAI*, pages 134–141. Morgan Kaufmann, 1999. URL: https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=162&proceeding_id=15.

23 Constantinos Daskalakis, Nishanth Dikkala, and Gautam Kamath. Testing ising models. In *SODA*, pages 1989–2007. SIAM, 2018. doi:10.1137/1.9781611975031.130.

24 Constantinos Daskalakis, Gautam Kamath, and John Wright. Which distribution distances are sublinearly testable? In *SODA*, pages 2747–2764. SIAM, 2018. doi:10.1137/1.9781611975031.175.

25 Constantinos Daskalakis and Qinxuan Pan. Square hellinger subadditivity for bayesian networks and its applications to identity testing. In *COLT*, volume 65 of *Proceedings of Machine Learning Research*, pages 697–703. PMLR, 2017. URL: http://proceedings.mlr.press/v65/daskalakis17a.html.

26 Ilias Diakonikolas, Themis Gouleakis, Daniel M. Kane, John Peebles, and Eric Price. Optimal testing of discrete distributions with high probability. In *STOC*, pages 542–555. ACM, 2021. doi:10.1145/3406325.3450997.

27 Ilias Diakonikolas, Themis Gouleakis, John Peebles, and Eric Price. Sample-optimal identity testing with high probability. In *ICALP*, LIPIcs, pages 41:1–41:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.41.

28 Ilias Diakonikolas and Daniel M. Kane. A new approach for testing properties of discrete distributions. In *FOCS*, pages 685–694. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.78.

29 Oded Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017. doi:10.1017/9781108135252.

30 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998. doi:10.1145/285055.285060.

31 Jiantao Jiao, Kartik Venkat, Yanjun Han, and Tsachy Weissman. Minimax estimation of functionals of discrete distributions. *IEEE Trans. Inf. Theory*, 61(5):2835–2885, 2015. doi:10.1109/TIT.2015.2412945.

32 Liam Paninski. A coincidence-based test for uniformity given very sparsely sampled discrete data. *IEEE Trans. Inf. Theory*, 54(10):4750–4755, 2008. doi:10.1109/TIT.2008.928987.

33 Yury Polyanskiy and Yihong Wu. *Information Theory: From Coding to Learning*. Cambridge University Press, 2025.

34 Ronitt Rubinfeld. Taming big probability distributions. *XRDS*, 19(1):24–28, 2012. doi:10.1145/2331042.2331052.

35 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.*, 25(2):252–271, 1996. doi:10.1137/S0097539793255151.

36 Igal Sason and Sergio Verdú. f-divergence inequalities. *IEEE Trans. Inf. Theory*, 62(11):5973–6006, 2016. doi:10.1109/TIT.2016.2603151.

37 Jan Seyfried, Sayantan Sen, and Marco Tomamichel. Testing (conditional) mutual information. *Electron. Colloquium Comput. Complex.*, TR25, 2025. URL: https://eccc.weizmann.ac.il/report/2025/076.

38 Flemming Topsøe. Some inequalities for information divergence and related measures of discrimination. *IEEE Trans. Inf. Theory*, 46(4):1602–1609, 2000. doi:10.1109/18.850703.

39 Gregory Valiant and Paul Valiant. Estimating the unseen: An n/log(n)-sample estimator for entropy and support size, shown optimal via new clts. In *STOC*, pages 685–694. ACM, 2011. doi:10.1145/1993636.1993727.

40 Gregory Valiant and Paul Valiant. An automatic inequality prover and instance optimal identity testing. In *FOCS*, pages 51–60. IEEE Computer Society, 2014. doi:10.1109/FOCS.2014.14.

41 Yihong Wu and Pengkun Yang. Optimal entropy estimation on large alphabets via best polynomial approximation. In *ISIT*, pages 824–828. IEEE, 2015. doi:10.1109/ISIT.2015.7282570.

42 Yihong Wu and Pengkun Yang. Minimax rates of entropy estimation on large alphabets via best polynomial approximation. *IEEE Trans. Inf. Theory*, 2016. doi:10.1109/TIT.2016.2548468.

A Deferred Proofs

▶ **Fact 5.** *Let X be a Poisson random variable with parameter λ, and $f\colon \mathbb{R} \to \mathbb{R}$ be any function, then, we have that $\mathbb{E}[(X)_m \cdot f(X)] = \lambda^m \cdot \mathbb{E}[f(X+m)]$. In particular, choosing $f \equiv 1$, we have that $\mathbb{E}[(X)_m] = \lambda^m$ for all $m \geqslant 0$.*

Proof. We first show that $\mathbb{E}[X \cdot f(X)] = \lambda \cdot \mathbb{E}[f(X+1)]$ by standard Poisson calculations:

$$\mathbb{E}[X \cdot f(X)] = \sum_{k=0} \frac{\lambda^k e^{-\lambda}}{k!} \cdot k \cdot f(k) = \lambda \cdot \sum_{k=1} \frac{\lambda^{k-1} e^{-\lambda}}{(k-1)!} \cdot f(k) = \lambda \cdot \sum_{k=0} \frac{\lambda^k e^{-\lambda}}{k!} f(k+1)$$
$$= \lambda \cdot \mathbb{E}[f(X+1)]$$

The proof is then completed by induction, by noting that

$$\begin{aligned}\mathbb{E}[(X)_m \cdot f(X)] &= \mathbb{E}[X \cdot (X-1)_{m-1} \cdot f(X)] \\ &= \lambda \cdot \mathbb{E}[(X-1+1)_{m-1} \cdot f(X+1)] \\ &= \lambda^2 \cdot \mathbb{E}[(X)_{m-2} \cdot f(X+2)] \\ &= \dots \\ &= \lambda^m \cdot \mathbb{E}[f(X+m)].\end{aligned}$$

◀

On Computing Total Variation Distance Between Mixtures of Product Distributions

Weiming Feng
School of Computing and Data Science, The University of Hong Kong, China

Yucheng Fu
School of Computing and Data Science, The University of Hong Kong, China

Minji Yang
School of Computing and Data Science, The University of Hong Kong, China

Anqi Zhang
The Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China

Abstract

We study the problem of approximating the total variation distance between two mixtures of product distributions over an n-dimensional discrete domain. Given two mixtures $\mathbb{P}$ and $\mathbb{Q}$ with k_1 and k_2 product distributions over $[q]^n$, respectively, we give a randomized algorithm that approximates $d_{\mathrm{TV}}(\mathbb{P},\mathbb{Q})$ within a multiplicative error of $(1\pm\varepsilon)$ in time $\mathrm{poly}((nq)^{k_1+k_2},1/\varepsilon)$. We also study the special case of mixtures of Boolean subcubes over $\{0,1\}^n$. For this class, we give a deterministic algorithm that exactly computes the total variation distance in time $\mathrm{poly}(n,2^{O(k_1+k_2)})$, and show that exact computation is #P-hard when $k_1+k_2=\Theta(n)$.

2012 ACM Subject Classification Theory of computation → Randomness, geometry and discrete structures

Keywords and phrases Randomized algorithm, Total variation distance

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.51

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2605.03839

Funding *Weiming Feng*: ECS grant 27202725 from Hong Kong RGC.

Acknowledgements We thank Arnab Bhattacharyya and Guy Van den Broeck for bringing the problem to our attention.

1 Introduction

Let $\mathbb{P}$ and $\mathbb{Q}$ be two discrete distributions over a sample space Ω. The *total variation distance (TV-distance)* between $\mathbb{P}$ and $\mathbb{Q}$ is defined as

$$d_{\mathrm{TV}}(\mathbb{P},\mathbb{Q})=\frac{1}{2}\sum_{\omega\in\Omega}|\mathbb{P}(\omega)-\mathbb{Q}(\omega)|=\sum_{\omega\in\Omega}\max\{0,\mathbb{P}(\omega)-\mathbb{Q}(\omega)\}.$$

The TV-distance is one of the most fundamental metrics for measuring the difference between distributions, as it characterizes the optimal distinguishability between $\mathbb{P}$ and $\mathbb{Q}$. Computing the TV-distance is therefore a basic problem arising in learning and testing.

A line of research has investigated the problem of *approximating* the TV-distance between two high-dimensional distributions, which is particularly interesting when the two distributions admit *succinct representations*. Early works [22, 8, 10, 19, 5] studied the algorithms and complexity of approximating the TV-distance with additive error. More recently, [2] established that even for two product distributions, the *exact computation* of the


© Weiming Feng, Yucheng Fu, Minji Yang, and Anqi Zhang;
licensed under Creative Commons License CC-BY 4.0

TV-distance is #P-hard. In the same paper, the authors initiated the study of approximating the TV-distance with *relative error*, a more challenging task than the classical additive-error approximation. Subsequently, polynomial-time approximation algorithms have been developed for product distributions [2, 12, 14], as well as for many high-dimensional distributions defined by *local interactions*, including Markov chains [14], Bayesian networks [3], and undirected graphical models [13].

Beyond local interactions, it is natural to consider more general high-dimensional distributions with inherently *non-local structure*. A fundamental and extensively studied example is given by *mixtures of product distributions*. Let $[q] = \{1, 2, \ldots, q\}$ and let $\Omega = [q]^n$ denote the n-dimensional sample space. Fix two integers $k_1, k_2 \geq 1$. For each $s \in [k_1]$, let $\mathbb{P}^{(s)} = \bigotimes_{i=1}^n \mathbb{P}_i^{(s)}$ be a product distribution over Ω, where $\mathbb{P}_i^{(s)}$ denotes the marginal on the i-th coordinate, so that the coordinates are mutually independent under $\mathbb{P}^{(s)}$. Let $\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k_1)}$ be mixing weights satisfying $\sum_{s=1}^{k_1} \alpha^{(s)} = 1$. The resulting mixture distribution is written as

$$\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}.$$

More explicitly, $\mathbb{P}$ is the distribution over $\Omega = [q]^n$ given by

$$\forall \omega \in \Omega, \quad \mathbb{P}(\omega) = \sum_{s=1}^{k_1} \alpha^{(s)} \mathbb{P}^{(s)}(\omega) = \sum_{s=1}^{k_1} \alpha^{(s)} \prod_{i=1}^{n} \mathbb{P}_i^{(s)}(\omega_i), \tag{1}$$

Mixtures of product distributions form a fundamental class of *latent-variable models* and have been extensively studied in learning theory [11, 17, 9, 15, 16]. To draw a random sample $X \sim \mathbb{P}$, one first samples a *hidden* component index $S \in [k_1]$ according to $\mathbf{Pr}[S = s] = \alpha^{(s)}$, and then samples $X \sim \mathbb{P}^{(S)} = \bigotimes_{i=1}^n \mathbb{P}_i^{(S)}$ from the product distribution $\mathbb{P}^{(S)}$. Although each component $\mathbb{P}^{(s)}$ is itself a product distribution, the hidden index S induces a highly *non-local dependency* that simultaneously couples all coordinates of the sample $X = (X_1, X_2, \ldots, X_n)$, which cannot be captured by any sparse local structure.

In this paper, we study the problem of *approximating* the TV-distance between two mixtures of product distributions to within an arbitrarily small relative error, formally stated as follows.

▶ **Problem 1.** *Approximate the TV-distance between two mixtures of product distributions.*

- ***Input****: An error bound $\varepsilon > 0$ and two mixtures of product distributions over $\Omega = [q]^n$:*

$$\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)} \text{ and } \mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}_i^{(t)}.$$

  The distribution $\mathbb{P}$ is specified by the k_1 mixing weights $\alpha^{(1)}, \alpha^{(2)}, \ldots, \alpha^{(k_1)}$ together with the k_1 product distributions $\mathbb{P}^{(1)}, \mathbb{P}^{(2)}, \ldots, \mathbb{P}^{(k_1)}$, where each $\mathbb{P}^{(s)}$ is described by its n marginals $\mathbb{P}_1^{(s)}, \mathbb{P}_2^{(s)}, \ldots, \mathbb{P}_n^{(s)}$ over $[q]$. The distribution $\mathbb{Q}$ is given analogously. The total input size of the two mixtures is $O((k_1 + k_2)nq)$.
- ***Output****: A number $\hat{d}$ such that*

$$(1 - \varepsilon) d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}) \leq \hat{d} \leq (1 + \varepsilon) d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}).$$

Due to the non-local nature of these dependencies among all coordinates, existing techniques for TV-distance approximation cannot be directly applied to mixtures of product distributions. The only prior progress is a polynomial-time algorithm for *deciding* whether

$d_{\mathrm{TV}}(\mathbb{P},\mathbb{Q}) = 0$ or $d_{\mathrm{TV}}(\mathbb{P},\mathbb{Q}) > 0$ [4], leaving the approximation problem open. We partially resolve this open question by giving an FPRAS for the TV-distance between two mixtures of product distributions, provided that the numbers of components k_1 and k_2 are constants.

▶ **Theorem 2.** *Let $k_1, k_2 \geq 1$ be constants. There exists an FPRAS for Problem 1: given any $\varepsilon > 0$ and any pair of mixtures of product distributions $\mathbb{P},\mathbb{Q}$ over $[q]^n$, where $\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}$ has k_1 components and $\mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}_i^{(t)}$ has k_2 components, it solves the problem in time $O(\frac{q^2(nq)^{2(k_1+k_2-1)}}{\varepsilon^2})$ with success probability at least 99%.*

Our algorithm is a Monte Carlo algorithm based on a carefully designed estimator, and we provide an overview of it in Section 2. Since the number of components $k_1 + k_2$ appears as an exponent in the running time, we require $k_1 + k_2$ to be a constant. A similar dependence on the number of components also arises in the running time of certain learning tasks for mixtures of product distributions [11, 9, 15]. We leave it as an open question to understand the computational complexity when the number of components grows with n.

Our second result concerns mixtures of Boolean subcubes, a special case of mixtures of product distributions. A distribution $\mathbb{P} = \sum_{s=1}^{k} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}$ over $\{0,1\}^n$ is called a mixture of Boolean subcubes if $\mathbb{P}_i^{(s)}(1) \in \{0, \frac{1}{2}, 1\}$ for every $s \in [k]$ and $i \in [n]$, where $\mathbb{P}_i^{(s)}$ denotes the marginal distribution of the i-th coordinate in the s-th component. Equivalently, for each component s, the product distribution $\mathbb{P}^{(s)}$ fixes every coordinate i with $\mathbb{P}_i^{(s)}(1) = 1$ to 1, fixes every coordinate i with $\mathbb{P}_i^{(s)}(1) = 0$ to 0, and is uniform on the remaining coordinates. Mixtures of Boolean subcubes have been extensively studied in learning theory [9, 6, 7] and capture many natural distributions; for instance, the uniform distribution over the satisfying assignments of a decision tree is a mixture of Boolean subcubes [11].

In contrast to the general case, where we only obtain an FPRAS when $k_1 + k_2 = O(1)$, we show that the TV-distance between two mixtures of Boolean subcubes $\mathbb{P}$ and $\mathbb{Q}$ can be computed *exactly* in time polynomial in the dimension n, provided that $k_1 + k_2 = O(\log n)$.

▶ **Problem 3.** *Compute exactly the total variation distance between two mixtures of Boolean subcubes $\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}$ and $\mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}_i^{(t)}$.*

- ***Input**: two mixtures of Boolean subcubes, given in the same format as in Problem 1.*
- ***Output**: the* exact *value of $d_{\mathrm{TV}}(\mathbb{P},\mathbb{Q})$*

▶ **Theorem 4.** *There exists a deterministic algorithm for Problem 3: given any pair of mixtures of Boolean subcubes $\mathbb{P},\mathbb{Q}$ over $\{0,1\}^n$, where $\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}$ has k_1 components and $\mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}_i^{(t)}$ has k_2 components, it solves the problem in $O((k_1+k_2)3^{k_1+k_2} \cdot n)$ time.*

The running time in Theorem 4 is polynomial in n whenever $k_1 + k_2 = O(\log n)$, yielding a deterministic polynomial-time algorithm in this regime. This does not contradict the #P-hardness of exact TV-distance computation established in [2]: their hardness already holds for two product distributions with arbitrary marginals, whereas our algorithm crucially exploits the restriction that each marginal of a Boolean subcube takes value in $\{0, \frac{1}{2}, 1\}$.

The idea of the algorithm is as follows. For each component $\mathbb{P}^{(s)}$ (and similarly for each $\mathbb{Q}^{(t)}$), the value of $\mathbb{P}^{(s)}(\omega)$ is either 0 or 2^{-r}, where r is the number of coordinates $i \in [n]$ with $\mathbb{P}_i^{(s)}(1) = \frac{1}{2}$. Indeed, if some coordinate is fixed to be 1 (or 0) but $\omega_i = 0$ (or 1), then $\mathbb{P}^{(s)}(\omega) = 0$; otherwise, each free coordinate contributes a factor of 1/2, so $\mathbb{P}^{(s)}(\omega) = 2^{-r}$. Thus, although there are 2^n configurations $\omega \in \{0,1\}^n$, the possible values of $|\mathbb{P}(\omega) - \mathbb{Q}(\omega)|$ are at most $2^{k_1+k_2}$. Our algorithm enumerates these possible values of $|\mathbb{P}(\omega) - \mathbb{Q}(\omega)|$ and counts the number of configurations $\omega \in \{0,1\}^n$ that achieve each one. The details are given in Section 5.

Finally, we complement this positive result by showing that the dependence on the number of components cannot be removed entirely: when the number of components grows linearly with n, exact computation becomes #P-hard.

▶ **Theorem 5.** *If the number of mixture components is $k_1 + k_2 = \Theta(n)$, then the exact computation of the total variation distance in Problem 3 is #P-hard.*

The theorem is proved by a reduction showing that if we can exactly compute the TV-distance between two mixtures of Boolean subcubes with $k_1 + k_2 = \Theta(n)$ in polynomial time, then we can also exactly compute #3SAT in polynomial time. The detailed proof is in Section 6.

Open problems. As discussed after Theorem 2, the dependence on the number of components in the running time is not yet well understood. [4] gave an algorithm that decides whether $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}) = 0$ in time $\mathrm{poly}(n, k_1 + k_2)$, whereas the running time of our approximation algorithm depends exponentially on $k_1 + k_2$. The analogous regime $k_1 + k_2 = \omega(1)$ is also not well understood for learning tasks on mixtures of product distributions, where obtaining algorithms with polynomial dependence on the number of components remains a central challenge [11, 9, 15]. A natural question is whether one can approximate $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q})$ in time polynomial in $k_1 + k_2$, or prove a hardness result ruling this out when the number of components grows with n. As a first step toward the latter, Theorem 5 already shows that exact computation is #P-hard when $k_1 + k_2 = \Theta(n)$, and one direction is to extend its proof to rule out approximation as well.

One can ask analogous TV-distance approximation questions for other structured mixture models. Natural candidates include mixtures of Gaussians [1, 21], mixtures of Markov chains and MDPs [18]. These models also give succinct descriptions of high-dimensional distributions and are widely studied in machine learning, but their dependence structures are more complex than that of mixtures of product distributions. More generally, it would be interesting to understand the complexity of computing or approximating the TV-distance between trajectory distributions induced by reward-mixing Markov decision processes (RMMDPs). RMMDPs use latent mixture structure to model the reward context, and TV-distance bounds between trajectory distributions have been used to learn near-optimal policies [20]. The problem of computing the TV-distance between RMMDPs appears to require new ideas.

At last, our algorithm for mixtures of product distributions is randomized. It remains open to design a deterministic algorithm for approximating the TV-distance between mixtures of product distributions. Currently, deterministic algorithms are only known for approximating the TV-distance between product distributions [14, 2].

2 Technical Overview

We present the main ideas behind the algorithm in Theorem 2. An obstacle for approximating the TV-distance is that the TV-distance itself can be very small. To see the reason, for two distributions $\mathbb{P}$ and $\mathbb{Q}$ over the sample space Ω, where $\mathbb{Q}$ is absolutely continuous w.r.t. $\mathbb{P}$, one can write

$$d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}) = \sum_{\omega \in \Omega} \mathbb{P}(\omega) \max\left\{0, 1 - \frac{\mathbb{Q}(\omega)}{\mathbb{P}(\omega)}\right\} = \mathbf{E}_{\omega \sim \mathbb{P}}\left[g(\omega)\right],$$

where $g(\omega) := \max\left\{0, 1 - \frac{\mathbb{Q}(\omega)}{\mathbb{P}(\omega)}\right\}$. We can efficiently sample ω and compute $g(\omega)$, so that a direct Monte Carlo estimate of $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}) = \mathbf{E}_{\omega \sim \mathbb{P}}\left[g(\omega)\right]$ can only achieve *additive* error in polynomial time. However, since the expectation $\mathbf{E}_{\omega \sim \mathbb{P}}\left[g(\omega)\right]$ itself can be exponentially small, this simple approach is not sufficient to achieve a *relative* error approximation required by Problem 1.

2.1 Approximating the TV-distance via a coarse coupling

A framework for approximating the TV-distance was proposed in [12] for two product distributions (i.e., $k_1 = k_2 = 1$ in our setting). It can be generalized to any two distributions $\mathbb{P}$ and $\mathbb{Q}$. Instead of approximating $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q})$ directly, they estimated the normalized quantity $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}) / \tilde{d}$, where $\tilde{d}$ is a polynomial-time computable *coarse* estimator. Moreover, if the ratio $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}) / \tilde{d}$ is lower bounded by $1/\mathrm{poly}(n)$, one can try to achieve a relative-error approximation to the ratio in polynomial time. Hence, we can recover $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q})$ by multiplying $\tilde{d}$ by the estimate of the ratio.

The coarse estimator $\tilde{d}$ can be constructed via a *coupling*. For two distributions $\mathbb{P}$ and $\mathbb{Q}$ over Ω, a coupling is a pair of joint random variables (X, Y) such that $X \sim \mathbb{P}$ and $Y \sim \mathbb{Q}$. It is well known that $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}) = \mathbf{Pr}_{\mathcal{O}}[X' \neq Y']$, where $\mathcal{O} = (X', Y')$ is the *optimal* coupling between $\mathbb{P}$ and $\mathbb{Q}$. Suppose we can construct a coupling $\mathcal{C} = (X, Y)$ that is not far from the optimal coupling $\mathcal{O}$. Formally, $\mathbf{Pr}_{\mathcal{O}}[X' \neq Y'] = \gamma \cdot \mathbf{Pr}_{\mathcal{C}}[X \neq Y]$ for some $\gamma = 1/\mathrm{poly}(n)$. Then $\tilde{d} = \mathbf{Pr}_{\mathcal{C}}[X \neq Y]$ is a coarse estimator of $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q})$. To estimate the ratio $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}) / \tilde{d}$, define a distribution π:

$$\omega \in \Omega, \quad \pi(\omega) = \mathbf{Pr}_{\mathcal{C}}[X = \omega \mid X \neq Y] \tag{2}$$

and define function f such that

$$\omega \in \Omega, \quad f(\omega) = \frac{\max\{0, \mathbb{P}(\omega) - \mathbb{Q}(\omega)\}}{\mathbf{Pr}_{\mathcal{C}}[X = \omega \wedge X \neq Y]}.$$

We can verify that $\mathbf{E}_{\pi}[f] = \frac{d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q})}{\mathbf{Pr}_{\mathcal{C}}[X \neq Y]} = \frac{d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q})}{\tilde{d}} = \gamma \geq \frac{1}{\mathrm{poly}(n)}$. Using the approach described above, if we can perform the following operations in polynomial time

(a) sample $\omega \sim \pi$;
(b) compute $f(\omega)$ for any ω;
(c) compute $\mathbf{Pr}_{\mathcal{C}}[X \neq Y]$;

then we can get a relative-error approximation of $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q})$. Specifically, we draw $T = \mathrm{poly}(\frac{1}{\gamma}) = \mathrm{poly}(n)$ independent samples $\omega_1, \ldots, \omega_T \sim \pi$ and compute $\bar{f} = \frac{1}{T} \sum_{i=1}^{T} f(\omega_i)$. By Chebyshev's inequality, with high probability, $\bar{f} \cdot \mathbf{Pr}_{\mathcal{C}}[X \neq Y]$ is an estimate of $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q})$ within relative error.

2.2 Recursive coupling for mixtures of product distributions

It remains to construct a coupling that supports the three operations above. For product distributions, [12] used a coordinate-wise greedy coupling $\mathcal{C} = (X, Y)$, which couples each coordinate optimally and independently, and showed that $1/n \leq \mathbf{E}_{\pi}[f] \leq 1$. This approach does not directly extend to mixtures of product distributions, where all the coordinates are *correlated*, so one cannot optimally couple all coordinates simultaneously.

Our main technical contribution in this part is a new *explicit recursive coupling* for mixtures of product distributions, together with an algorithmic implementation of the three operations above. At a high level, the recursive structure draws inspiration from [20], who introduced a recursive argument to bound the TV-distance between two mixtures of product distributions. Their analysis, however, is non-constructive: it yields a *fixed-error* bound on the TV-distance, but does not by itself provide the algorithmic interface required here. To obtain a relative-error approximation with *arbitrary* ε, we need a coupling whose failure probability can be computed, and whose failure-conditioned distribution can be sampled and evaluated. We design such a coupling tailored to mixtures of product distributions, adapt the recursive analysis of [20] to this explicit construction, and show that the three operations in Item a, Item b, and Item c can all be implemented in polynomial time.

We construct a coupling between two mixtures of product distributions $\mathbb{P}$ and $\mathbb{Q}$, where

$$\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)} \quad \text{and} \quad \mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}_i^{(t)}.$$

Although coupling the first coordinate would bias the marginal distribution of the remaining coordinates, we can still extract the common parts of $\mathbb{P}_1^{(s)}$ and $\mathbb{Q}_1^{(t)}$, and reduce the problem to coupling *another* mixture of product distributions on the remaining coordinates. For each $c \in [q]$, we first compute a lower bound for each active mixture component:

$$\ell(c) \triangleq \min \left\{ \min_{s \in [k_1]: \alpha^{(s)} > 0} \mathbb{P}_1^{(s)}(c), \min_{t \in [k_2]: \beta^{(t)} > 0} \mathbb{Q}_1^{(t)}(c) \right\}, \tag{3}$$

where $\mathbb{P}_1^{(s)}$ (resp. $\mathbb{Q}_1^{(t)}$) is the marginal distribution of the first coordinate of the s-th component of $\mathbb{P}^{(s)}$ (resp. $\mathbb{Q}^{(t)}$). For $\omega \in [q]^n$ with $\omega_1 = c$, we can write the probability of ω as

$$\begin{aligned}
\mathbb{P}(\omega) &= \ell(c) \sum_{s=1}^{k_1} \alpha^{(s)} \prod_{i=2}^{n} \mathbb{P}_i^{(s)}(\omega_i) + \sum_{s=1}^{k_1} \alpha^{(s)} \left(\mathbb{P}_1^{(s)}(c) - \ell(c) \right) \prod_{i=2}^{n} \mathbb{P}_i^{(s)}(\omega_i) \\
&= \ell(c) \underbrace{\sum_{s=1}^{k_1} \alpha^{(s)} \prod_{i=2}^{n} \mathbb{P}_i^{(s)}(\omega_i)}_{A_{\mathbb{P}}} + (\mathbb{P}_1(c) - \ell(c)) \underbrace{\sum_{s=1}^{k_1} \alpha_c^{(s)} \prod_{i=2}^{n} \mathbb{P}_i^{(s)}(\omega_i)}_{B_{\mathbb{P},c}},
\end{aligned}$$

where $\alpha_c^{(s)} \triangleq \frac{\alpha^{(s)} \left(\mathbb{P}_1^{(s)}(c) - \ell(c) \right)}{(\mathbb{P}_1(c) - \ell(c))}$ is a valid distribution over $s \in [k_1]$. Note that $\mathbb{P}_1$ in the denominator is the marginal distribution of the mixture $\mathbb{P}$ over the first coordinate. Similarly, let $\beta_c^{(t)} \triangleq \frac{\beta^{(t)} \left(\mathbb{Q}_1^{(t)}(c) - \ell(c) \right)}{(\mathbb{Q}_1(c) - \ell(c))}$ and

$$\mathbb{Q}(\omega) = \ell(c) \underbrace{\sum_{t=1}^{k_2} \beta^{(t)} \prod_{i=2}^{n} \mathbb{Q}_i^{(t)}(\omega_i)}_{A_{\mathbb{Q}}} + (\mathbb{Q}_1(c) - \ell(c)) \underbrace{\sum_{t=1}^{k_2} \beta_c^{(t)} \prod_{i=2}^{n} \mathbb{Q}_i^{(t)}(\omega_i)}_{B_{\mathbb{Q},c}}.$$

Each of $A_{\mathbb{P}}, A_{\mathbb{Q}}, B_{\mathbb{P},c}$, and $B_{\mathbb{Q},c}$ is a mixture of product distributions over the remaining coordinates. The component distributions in $A_{\mathbb{P}}$ and $B_{\mathbb{P},c}$ are obtained from the components of $\mathbb{P}$ by removing the first coordinate; the same relation holds between $A_{\mathbb{Q}}, B_{\mathbb{Q},c}$ and $\mathbb{Q}$. However, the mixing weights may be different from those in $\mathbb{P}$ and $\mathbb{Q}$.

We use the above decomposition to construct a coupling (X, Y) between $\mathbb{P}$ and $\mathbb{Q}$. We first couple the first coordinate of X and Y. For each $c \in [q]$, with probability $\ell(c)$, we set $X_1 = Y_1 = c$ and then couple $A_{\mathbb{P}}$ and $A_{\mathbb{Q}}$ *recursively* for the remaining coordinates. For each $c \in [q]$, with probability $\min\{\mathbb{P}_1(c) - \ell(c), \mathbb{Q}_1(c) - \ell(c)\}$, we set $X_1 = Y_1 = c$ and couple $B_{\mathbb{P},c}$ and $B_{\mathbb{Q},c}$ *recursively* for the remaining coordinates. For the remaining probability mass[1] of the first coordinate, there is no common part with $X_1 = Y_1 = c$ for any $c \in [q]$. We arbitrarily couple this remaining mass; it must hold that $X_1 \neq Y_1$, and hence $X \neq Y$. We call this case the *coupling failure* case. Suppose that after coupling the first coordinate, we have $X_1 = c \neq c' = Y_1$ for some $c, c' \in [q]$. We then independently sample the remaining coordinates according to $B_{\mathbb{P},c}$ and $B_{\mathbb{Q},c'}$, respectively.

[1] Specifically, the remaining probability for $X_1 = c$ is $\max\{0, \mathbb{P}_1(c) - \mathbb{Q}_1(c)\}$ and the remaining probability for $Y_1 = c$ is $\max\{0, \mathbb{Q}_1(c) - \mathbb{P}_1(c)\}$.

The above recursive process terminates either when coupling fails at some coordinate or when it reaches the last coordinate. Denote the resulting recursive coupling by $\mathcal{C}_{\mathrm{RC}}$. Adapting the recursive TV-distance analysis of [20] to this explicit coupling, we show that $\mathcal{C}_{\mathrm{RC}}$ is not far from the optimal coupling:

$$\mathbf{Pr}_{\mathcal{C}_{\mathrm{RC}}}[X \neq Y] \leq (4nq)^{k_1+k_2-1} \cdot d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}).$$

Therefore, we set $\gamma = (4nq)^{-k_1-k_2+1}$, which is inverse-polynomial in n and q when k_1 and k_2 are constants.

2.3 Implementation of three operations

We now give polynomial-time algorithms for the three operations in Item a, Item b, and Item c. We create a state space $\mathcal{S}$ to keep track of the recursive coupling process. A state $S \in \mathcal{S}$ is a tuple $S = (i, \bar{\alpha}, \bar{\beta})$, where $i \in [n]$ means we need to couple the remaining coordinates $i, i+1, \ldots, n$, and $\bar{\alpha} = (\bar{\alpha}^{(s)})_{s\in[k_1]}$ and $\bar{\beta} = (\bar{\beta}^{(t)})_{t\in[k_2]}$ are the *current* mixing weights for each mixture component. Section 2.2 describes the recursive coupling starting from the root state $(1, \alpha, \beta)$, and hence defines the quantities $A_{\mathbb{P}}, A_{\mathbb{Q}}, B_{\mathbb{P},c}, B_{\mathbb{Q},c}$ and $\ell(c)$ in that setting. For the implementation, we use the same decomposition at an arbitrary state $S = (i, \bar{\alpha}, \bar{\beta})$, replacing the first coordinate by the current coordinate i and the original mixing weights by the current weights $\bar{\alpha}, \bar{\beta}$. Thus,

$$\ell(c) = \min\left\{ \min_{s\in[k_1]:\bar{\alpha}^{(s)}>0} \mathbb{P}_i^{(s)}(c), \min_{t\in[k_2]:\bar{\beta}^{(t)}>0} \mathbb{Q}_i^{(t)}(c) \right\}.$$

Similarly, the updated weights $\bar{\alpha}_c = (\bar{\alpha}_c^{(s)})_{s\in[k_1]}$ in the $B_{\mathbb{P},c}$ branch are defined by

$$\forall s \in [k_1], \bar{\alpha}_c^{(s)} = \bar{\alpha}^{(s)} \cdot \frac{\mathbb{P}_i^{(s)}(c) - \ell(c)}{\bar{\mathbb{P}}_i(c) - \ell(c)} \text{ where } \bar{\mathbb{P}}_i(c) = \sum_{s\in[k_1]} \bar{\alpha}^{(s)} \mathbb{P}_i^{(s)}(c).$$

The $\bar{\beta}_c$ for the $B_{\mathbb{Q},c}$ is analogous. There is also a failure state $\perp$. If we couple $X_i \neq Y_i$ (coupling failure case), the state S transitions to the failure state $\perp$. If we couple $X_i = Y_i$ and then recursively couple $A_{\mathbb{P}}$ and $A_{\mathbb{Q}}$ for the remaining coordinates (we call it the good case A), the state S transitions to $S' = (i+1, \bar{\alpha}, \bar{\beta})$. If we couple $X_i = Y_i$ and then recursively couple $B_{\mathbb{P},c}$ and $B_{\mathbb{Q},c}$ for the remaining coordinates (we call it the good case B), the state S transitions to $S'_c = (i+1, \bar{\alpha}_c, \bar{\beta}_c)$.

The whole recursive coupling process is a random walk on the state space $\mathcal{S}$ starting from the root state $S_{\mathrm{root}} = (1, \alpha, \beta)$. Every non-failure state $S = (i, \bar{\alpha}, \bar{\beta}) \neq \perp$ can transition to a state in $\{\perp\} \cup \{S'\} \cup \{S'_c\}_{c\in[q]}$, and it transitions to some S'_c if and only if the good case B occurs. The key observation is that case B can occur at most $k_1 + k_2 - 2$ times in the whole recursive coupling process. By the definition of $\ell(c)$, $\ell(c)$ is equal to some $\mathbb{P}_i^{(s)}(c)$ or $\mathbb{Q}_i^{(t)}(c)$. Suppose $\ell(c) = \mathbb{P}_i^{(s)}(c)$. Then, the new mixing weights in S'_c satisfy $\bar{\alpha}_c^{(s)} = \bar{\alpha}^{(s)} \cdot \frac{\mathbb{P}_i^{(s)}(c) - \ell(c)}{\bar{\mathbb{P}}_i(c) - \ell(c)} = 0$. Thus, whenever case B occurs, the total number of positive mixing weights decreases by at least one. Since the root state has $k_1 + k_2$ positive mixing weights, case B can occur at most $k_1 + k_2 - 2$ times. Using this property, we can show that the total number of states is $|\mathcal{S}| = (nq)^{O(k_1+k_2)}$, which is polynomial in n and q.

Given a state S, it is easy to compute the transition probabilities to the next states. The whole transition graph forms a DAG with polynomial size. To sample $\omega \sim \pi$ in (2), it suffices to sample a trajectory starting from the root state, conditioned on the random walk reaching the failure state $\perp$. We first run a dynamic programming algorithm to compute the

probability of reaching $\perp$ from each state in $\mathcal{S}$. Then, we use these probabilities to sample the desired conditional trajectory. Similarly, using dynamic programming, we can compute $f(\omega)$ and $\mathbf{Pr}_{\mathcal{C}_{\mathrm{RC}}}[X \neq Y]$ in polynomial time.

▶ Remark 6 (Comparison with the coupling in [3]). Finally, it is worth comparing our coupling with the coupling used in [3]. They proposed a simpler coupling and applied it to approximating the TV-distance between two Bayesian networks. Given two n-dimensional distributions $\mathbb{P}$ and $\mathbb{Q}$ over $[q]^n$, they couple the coordinates one by one using the optimal coupling of the conditional marginals. Suppose the first $i-1$ coordinates of X and Y have been generated and coupled consistently such that $X_{1:i-1} = Y_{1:i-1} = \sigma \in [q]^{i-1}$. For the i-th coordinates X_i and Y_i, they use the optimal coupling of the conditional marginals $\mathbb{P}_i$ and $\mathbb{Q}_i$ given σ. The coupling terminates if $X_i \neq Y_i$ at some coordinate or when it reaches the last coordinate.

Their coupling is *different* from ours. Unlike Bayesian networks, for general mixtures of product distributions, the conditional marginals at the i-th coordinate require the *full information* of the prefix condition $\sigma \in [q]^{i-1}$. Since there are $O(q^n)$ possible prefix conditions σ, this coupling needs to consider exponentially many possible distributions for our problem. In our coupling, even when the good event $X_i = Y_i$ occurs, we further split it into two sub-cases, A and B. This splitting allows us to exploit the structure of mixtures of product distributions, so the total number of possible distributions appearing in our coupling is bounded by a polynomial.

3 A General Approach for Approximating the TV-Distance

When $k_1 = k_2 = 1$, the problem reduces to approximating the TV-distance between two product distributions P and Q over $\Omega = [q]^n$. This special case was studied in [12]. Although the algorithm there is tailored to product distributions, its underlying idea extends to a more general framework for approximating TV-distances for general discrete distributions.

Let $\mathbb{P}$ and $\mathbb{Q}$ be arbitrary discrete distributions over Ω. A *coupling* of $\mathbb{P}$ and $\mathbb{Q}$ is a pair of jointly distributed random variables (X, Y) such that $X \sim \mathbb{P}$ and $Y \sim \mathbb{Q}$. For every coupling $\mathcal{C}$ of $\mathbb{P}$ and $\mathbb{Q}$, the coupling inequality states that

$$d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}) \leq \mathbf{Pr}_{\mathcal{C}}[X \neq Y].$$

A coupling $\mathcal{C}$ is called *optimal* if equality holds, namely if $\mathbf{Pr}_{\mathcal{C}}[X \neq Y] = d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q})$. An optimal coupling exists for every pair of distributions $\mathbb{P}$ and $\mathbb{Q}$.

The following fact holds for every coupling $\mathcal{C}$ of $\mathbb{P}$ and $\mathbb{Q}$.

▶ **Fact 7.** *For any coupling $\mathcal{C}$ of $\mathbb{P}$ and $\mathbb{Q}$, it holds that*

$$\forall \omega \in \Omega, \quad \mathbf{Pr}_{\mathcal{C}}[X = \omega \wedge X \neq Y] \geq \max\{0, \mathbb{P}(\omega) - \mathbb{Q}(\omega)\}.$$

If $\mathcal{C}$ is an optimal coupling, then for all $\omega \in \Omega$,

$$\mathbf{Pr}_{\mathcal{C}}[X = \omega \wedge X \neq Y] = \max\{0, \mathbb{P}(\omega) - \mathbb{Q}(\omega)\}.$$

The proof can be found in the full version.

We now state an abstract version of the algorithm from [12], formulated for two arbitrary discrete distributions $\mathbb{P}$ and $\mathbb{Q}$. Let $\gamma, T_0, T_1, T_2, T_3 > 0$ be parameters.

▶ **Assumption 8.** *There exists a probability mass oracle that, for any $\omega \in \Omega$, returns $\mathbb{P}(\omega)$ and $\mathbb{Q}(\omega)$ in time T_0.*

▶ **Assumption 9.** *There exists a coupling $\mathcal{C}$ between $\mathbb{P}$ and $\mathbb{Q}$ such that $\frac{d_{\mathrm{TV}}(\mathbb{P},\mathbb{Q})}{\mathbf{Pr}_{\mathcal{C}}[X\neq Y]} \geq \gamma$, together with an oracle that can be preprocessed in time T_1 and supports the following query:*

- *Discrepancy query: return the value of $\mathbf{Pr}_{\mathcal{C}}[X \neq Y]$ in time $O(1)$.*

Furthermore, if $\mathbf{Pr}_{\mathcal{C}}[X \neq Y] > 0$, then the oracle also supports the following two queries:

- *Sampling query: draw an independent sample of X from the conditional distribution of X given $X \neq Y$ under $\mathcal{C}$ in time T_2. Equivalently, it returns $\omega \in \Omega$ with probability $\mathbf{Pr}_{\mathcal{C}}[X = \omega \mid X \neq Y]$.*
- *Evaluation query: given any $\omega \in \Omega$, return the value of $\mathbf{Pr}_{\mathcal{C}}[X = \omega \wedge X \neq Y]$ in time T_3.*

Under Assumption 8 and Assumption 9, the following theorem gives an algorithm for approximating the TV-distance between two *arbitrary* discrete distributions $\mathbb{P}$ and $\mathbb{Q}$.

▶ **Theorem 10** ([12]). *Suppose Assumption 8 and Assumption 9 hold for distributions $\mathbb{P}$ and $\mathbb{Q}$ with parameters $\gamma, T_0, T_1, T_2, T_3$. Then there exists a randomized algorithm such that, given any $\varepsilon > 0$, the parameters γ, and access to the two oracles above, it outputs a random number $\hat{d}$ in time $O(T_1 + \frac{T_0+T_2+T_3}{\gamma\varepsilon^2})$ satisfying $\mathbf{Pr}\left[(1-\varepsilon)d_{\mathrm{TV}}(\mathbb{P},\mathbb{Q}) \leq \hat{d} \leq (1+\varepsilon)d_{\mathrm{TV}}(\mathbb{P},\mathbb{Q})\right] \geq 99\%$.*

The proof can be found in the full version.

4 Algorithm for Mixtures of Product Distributions

To apply Theorem 10 to the problem of approximating the TV-distance between two mixtures of product distributions, we need to verify Assumption 8 and Assumption 9. By the definition of mixture of product distributions in (1), the first assumption holds with $T_0 = O(n(k_1+k_2))$. Our main task is to construct a coupling to verify the second assumption.

▶ **Lemma 11.** *Let $k_1, k_2 \geq 1$ be two constants. For any two mixtures of product distributions $\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}$ and $\mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}_i^{(t)}$, there exists a coupling $\mathcal{C}$ between $\mathbb{P}$ and $\mathbb{Q}$ such that Assumption 9 holds with parameters $\gamma = (4nq)^{-k_1-k_2+1}$, $T_1, T_2, T_3 = O(q^2(nq)^{k_1+k_2-1})$.*

Theorem 2 is a simple corollary of Theorem 10 and Lemma 11. The rest of this section is devoted to the proof of the lemma.

4.1 The recursive coupling for mixtures of product distributions

In this subsection, we construct a recursive coupling between two mixtures of product distributions $\mathbb{P}$ and $\mathbb{Q}$ in two steps. In Section 4.1.1, we first introduce a recursive procedure for drawing a sample from a single mixture $\mathbb{P}$. Building on this procedure, in Section 4.1.2 we then construct a recursive coupling between $\mathbb{P}$ and $\mathbb{Q}$. Our construction is inspired by the technique of [20], who develop a recursive analysis that bounds the total variation distance between two mixtures of product distributions up to a *fixed* error. Our main contribution is a new recursive coupling process, which is crucial for approximating the TV-distance with *arbitrary* relative error. In Section 4.2, we use this process to implement an oracle that answers the queries in Assumption 9 within polynomial time.

4.1.1 The recursive sampling process

We now describe a recursive sampling procedure that generates a sample X from the mixture of product distributions

$$\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}.$$

This sampling procedure will later serve as the basis for our coupling construction. Throughout the recursion, the component product distributions $\mathbb{P}^{(s)}$ remain fixed, while the mixing weights are updated from one recursive step to the next. We use $(\bar{\alpha}^{(s)})_{s=1}^{k_1}$ to denote the mixing weights at the current recursive step, to distinguish them from the original mixing weights $(\alpha^{(s)})_{s=1}^{k_1}$. At recursion depth j, let

$$\bar{\mathbb{P}} = \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \bigotimes_{i=j}^{n} \mathbb{P}_i^{(s)}$$

be the current mixture over $[q]^{n-j+1}$. We consider the recursive coordinate-wise procedure

$$\text{RecursiveSample}\,(j, \bar{\alpha})\,,$$

where $j \in [n]$ is the current coordinate. At this point, the coordinates $X_1, \ldots, X_{j-1}$ have already been fixed, and the task is to generate X_j from the current mixture $\bar{\mathbb{P}}$ and then update the mixing weights for the next recursive step. To describe the procedure, we define a lower-bound selector.

▶ **Definition 12** (lower-bound selector). *A lower-bound selector $\ell^{\mathbb{P}}$ is a function $\ell^{\mathbb{P}}(j, \bar{\alpha}, c)$ defined for any coordinate $j \in [n]$, any mixing-weight vector $\bar{\alpha} = (\bar{\alpha}^{(s)})_{s=1}^{k_1}$, and any $c \in [q]$, satisfying:*

$$\min_{s \in [k_1] : \bar{\alpha}^{(s)} > 0} \mathbb{P}_j^{(s)}(c) \geq \ell^{\mathbb{P}}(j, \bar{\alpha}, c) \geq 0. \tag{4}$$

Here the minimum ranges over all components $s \in [k_1]$ with $\bar{\alpha}^{(s)} > 0$, namely over the components that are still active at the current recursive step. Thus $\ell^{\mathbb{P}}(j, \bar{\alpha}, c)$ is required to be a nonnegative lower bound on the marginal probability of seeing value c in coordinate j across all active components. The above definition does *not* specify a unique lower-bound selector $\ell^{\mathbb{P}}$. Any function satisfying (4) is valid. Let us fix an arbitrary lower-bound selector $\ell^{\mathbb{P}}$ satisfying (4). Our recursive sampling process in Section 4.1.1 works for any lower-bound selector $\ell^{\mathbb{P}}$. In Section 4.1.2, when constructing the coupling, we will specify a particular choice of $\ell^{\mathbb{P}}$.

For the current recursive state $(j, (\bar{\alpha}^{(s)})_{s=1}^{k_1})$, we first compute the lower bounds $\ell^{\mathbb{P}}(j, \bar{\alpha}, c)$ for all $c \in [q]$. Fix a value $c \in [q]$ and an assignment $\omega \in [q]^{n-j+1}$ satisfying $\omega_j = c$. The probability of ω under the current mixture $\bar{\mathbb{P}}$ can be written as

$$\bar{\mathbb{P}}(\omega) = \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \prod_{i=j}^{n} \mathbb{P}_i^{(s)}(\omega_i) = \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \mathbb{P}_j^{(s)}(c) \prod_{i=j+1}^{n} \mathbb{P}_i^{(s)}(\omega_i),$$

where the second equality uses the assumption $\omega_j = c$. By (4), it holds that $\mathbb{P}_j^{(s)}(c) \geq \ell^{\mathbb{P}}(j, \bar{\alpha}, c)$ for all $s \in [k_1]$ with $\bar{\alpha}^{(s)} > 0$. Hence, for each s with $\bar{\alpha}^{(s)} > 0$, we can decompose $\mathbb{P}_j^{(s)}(c)$ as two non-negative terms: $\ell^{\mathbb{P}}(j, \bar{\alpha}, c)$ and $\mathbb{P}_j^{(s)}(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c)$. We have the following decomposition:

$$\bar{\mathbb{P}}(\omega) = \ell^{\mathbb{P}}(j, \bar{\alpha}, c) \underbrace{\sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \prod_{i=j+1}^{n} \mathbb{P}_i^{(s)}(\omega_i)}_{A} + \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \left(\mathbb{P}_j^{(s)}(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c) \right) \prod_{i=j+1}^{n} \mathbb{P}_i^{(s)}(\omega_i). \tag{5}$$

The above summation enumerates all $s \in [k_1]$, which may contain inactive components s with $\bar{\alpha}^{(s)} = 0$. However, those inactive components only contribute 0 to the whole sum. In (5), the first term is the product of $\ell^{\mathbb{P}}(j, \bar{\alpha}, c)$ and A, where A is exactly the probability of the suffix $(\omega_i)_{i=j+1}^{n}$ under the current mixture $\sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \bigotimes_{i=j+1}^{n} \mathbb{P}_i^{(s)}$. To write the second term in a similar form, we introduce a modified mixing weight $\bar{\alpha}_{j,c}^{(s)}$ for all $s \in [k_1]$ as follows:

$$\bar{\alpha}_{j,c}^{(s)} \triangleq \begin{cases} \bar{\alpha}^{(s)}, & \bar{\mathbb{P}}_j(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c) = 0; \\ \bar{\alpha}^{(s)} \frac{\mathbb{P}_j^{(s)}(c) - \ell^{\mathbb{P}}(j,\bar{\alpha},c)}{\bar{\mathbb{P}}_j(c) - \ell^{\mathbb{P}}(j,\bar{\alpha},c)}, & \text{otherwise,} \end{cases} \quad \text{where } \bar{\mathbb{P}}_j(c) = \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \mathbb{P}_j^{(s)}(c).$$

We next verify that $\bar{\alpha}_{j,c}$ defines a valid mixing-weight vector. By Definition 12, $\mathbb{P}_j^{(s)}(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c) \geq 0$, and $\bar{\mathbb{P}}_j(c) = \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \mathbb{P}_j^{(s)}(c)$ is an average of the active values $\mathbb{P}_j^{(s)}(c)$, therefore $\bar{\mathbb{P}}_j(c) \geq \ell^{\mathbb{P}}(j, \bar{\alpha}, c)$. If $\bar{\mathbb{P}}_j(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c) = 0$, $\bar{\alpha}_{j,c} = \bar{\alpha}$ is a valid mixing-weight vector as long as $\bar{\alpha}$ is valid. Otherwise, when $\bar{\mathbb{P}}_j(c) > \ell^{\mathbb{P}}(j, \bar{\alpha}, c)$,

$$\sum_{s=1}^{k_1} \bar{\alpha}_{j,c}^{(s)} = \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \frac{\mathbb{P}_j^{(s)}(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c)}{\bar{\mathbb{P}}_j(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c)} = \frac{\sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \mathbb{P}_j^{(s)}(c)}{\bar{\mathbb{P}}_j(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c)} - \frac{\ell^{\mathbb{P}}(j, \bar{\alpha}, c)}{\bar{\mathbb{P}}_j(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c)} = 1.$$

Hence, for $\omega \in [q]^{n-j+1}$ with $\omega_j = c$, we can rewrite $\bar{\mathbb{P}}(\omega)$ as

$$\bar{\mathbb{P}}(\omega) = \ell^{\mathbb{P}}(j, \bar{\alpha}, c) \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \prod_{i=j+1}^{n} \mathbb{P}_i^{(s)}(\omega_i) + \left(\bar{\mathbb{P}}_j(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c)\right) \sum_{s=1}^{k_1} \bar{\alpha}_{j,c}^{(s)} \prod_{i=j+1}^{n} \mathbb{P}_i^{(s)}(\omega_i). \quad (6)$$

Equation (6) naturally suggests a recursive sampling rule. Given the state $(j, (\bar{\alpha}^{(s)})_{s=1}^{k_1})$, we first sample the current coordinate X_j, and then reduce the task of sampling the suffix $X_{j+1:n}$ to sampling from one of two smaller mixtures. Concretely, once the current coordinate is set to c, the suffix distribution is either $\sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \bigotimes_{i=j+1}^{n} \mathbb{P}_i^{(s)}$ or $\sum_{s=1}^{k_1} \bar{\alpha}_{j,c}^{(s)} \bigotimes_{i=j+1}^{n} \mathbb{P}_i^{(s)}$. The component distributions remain the same (except that their dimension is reduced by one), while the mixing weights may change from $\bar{\alpha}$ to $\bar{\alpha}_{j,c}$. Applying this rule recursively yields the sampling procedure shown in Algorithm 1.

In Algorithm 1, the lower-bound selector function $\ell^{\mathbb{P}}$ is treated as part of the input. The following lemma shows that the algorithm outputs a correct sample from $\mathbb{P}$ as long as $\ell^{\mathbb{P}}$ satisfies (4).

▶ **Lemma 13.** *Given a mixture of product distributions $\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}$ and an arbitrary lower-bound selector function $\ell^{\mathbb{P}}$ satisfying (4), the procedure RecursiveSample$(1, \alpha)$ in Algorithm 1 outputs a sample $X \sim \mathbb{P}$.*

Lemma 13 follows by induction on the recursion depth using (6). The proof can be found in the full version.

The same recursive procedure applies to any mixture of product distributions. In particular, we may use it to sample from $\bar{\mathbb{Q}} = \sum_{t=1}^{k_2} \bar{\beta}^{(t)} \bigotimes_{i=j}^{n} \mathbb{Q}_i^{(t)}$ by replacing k_1 with k_2, replacing $(\bar{\alpha}^{(s)})_{s=1}^{k_1}$ with $(\bar{\beta}^{(t)})_{t=1}^{k_2}$, replacing $\mathbb{P}^{(s)}$ with $\mathbb{Q}^{(t)}$, and using an arbitrary lower-bound selector $\ell^{\mathbb{Q}}(j, \bar{\beta}, c)$ satisfying

$$\min_{t \in [k_2]: \bar{\beta}^{(t)} > 0} \mathbb{Q}_j^{(t)}(c) \geq \ell^{\mathbb{Q}}(j, \bar{\beta}, c) \geq 0. \quad (7)$$

Lemma 13 also shows that by substituting P with Q and α with β, the procedure RecursiveSample$(1, \beta)$ outputs a sample $Y \sim \mathbb{Q}$.

Algorithm 1 Recursive sampling process for mixtures of product distributions.

1 RecursiveSample $(j, \bar{\alpha})$;
Parameter : an index $j \in [n+1]$ and mixing weights $\bar{\alpha}^{(1)}, \bar{\alpha}^{(2)}, \ldots, \bar{\alpha}^{(k_1)}$
Static Input : product distributions $\mathbb{P}^{(1)}, \mathbb{P}^{(2)}, \ldots, \mathbb{P}^{(k_1)}$ over $[q]^n$; an arbitrary lower-bound selector function $\ell^{\mathbb{P}}$ satisfying (4)
Output : a random sample $X = (X_j, X_{j+1}, \ldots, X_n) \sim \bar{\mathbb{P}} = \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \bigotimes_{i=j}^{n} \mathbb{P}_i^{(s)}$
2 **if** $j > n$ **then return** $\emptyset$;
3 Compute the probability lower bound $\ell^{\mathbb{P}}(j, \bar{\alpha}, c)$ for all $c \in [q]$;
4 Sample a random pair $(c, e) \in [q] \times \{0, 1\}$ such that for each $c \in [q]$, the probability of $(c, 0)$ is $\ell^{\mathbb{P}}(j, \bar{\alpha}, c)$ and the probability of $(c, 1)$ is $\bar{\mathbb{P}}_j(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c)$, where $\bar{\mathbb{P}}_j(c) = \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \mathbb{P}_j^{(s)}(c)$ is the marginal probability of coordinate j under $\bar{\mathbb{P}}$;
5 **if** $e = 0$ **then**
6 Sample $(X_{j+1}, X_{j+2}, \ldots, X_n) \leftarrow$ RecursiveSample$(j+1, \bar{\alpha})$;
7 **else**
8 Compute mixing weights $\bar{\alpha}_{j,c}^{(s)} = \bar{\alpha}^{(s)} \frac{\mathbb{P}_j^{(s)}(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c)}{\bar{\mathbb{P}}_j(c) - \ell^{\mathbb{P}}(j, \bar{\alpha}, c)}$ for all $s \in [k_1]$;
9 Sample $(X_{j+1}, X_{j+2}, \ldots, X_n) \leftarrow$ RecursiveSample$(j+1, \bar{\alpha}_{j,c})$;
10 **return** $(X_j = c, X_{j+1}, X_{j+2}, \ldots, X_n)$;

4.1.2 The recursive coupling process

We now define a coupling between two mixtures of product distributions

$$\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)} \qquad \text{and} \qquad \mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}_i^{(t)}.$$

At a high level, the coupling runs Algorithm 1 on both $\mathbb{P}$ and $\mathbb{Q}$ to generate samples $X \sim \mathbb{P}$ and $Y \sim \mathbb{Q}$, while sharing randomness between the two executions so that the resulting pair (X, Y) is a coupling of $\mathbb{P}$ and $\mathbb{Q}$.

We use RecursiveSample$(1, \alpha)$ to generate a sample $X \sim \mathbb{P}$ and RecursiveSample$(1, \beta)$ to generate a sample $Y \sim \mathbb{Q}$, and couple the two recursive procedures at the same time. Suppose that at some point the execution for $\mathbb{P}$ is at recursion state RecursiveSample$(j, \bar{\alpha})$ and the execution for $\mathbb{Q}$ is at recursion state RecursiveSample$(j, \bar{\beta})$, where $\bar{\alpha} = (\bar{\alpha}^{(s)})_{s=1}^{k_1}$ and $\bar{\beta} = (\bar{\beta}^{(t)})_{t=1}^{k_2}$ denote the current mixing weights and may differ from the original vectors α and β. We define a shared lower-bound selector $\ell(j, \bar{\alpha}, \bar{\beta}, c)$ as follows.

▶ **Definition 14.** *For any $j \in [n]$, $c \in [q]$, and valid mixing-weight vectors $\bar{\alpha} = (\bar{\alpha}^{(s)})_{s=1}^{k_1}$ and $\bar{\beta} = (\bar{\beta}^{(t)})_{t=1}^{k_2}$, define*

$$\ell\left(j, \bar{\alpha}, \bar{\beta}, c\right) \triangleq \min\left\{ \min_{s \in [k_1] : \bar{\alpha}^{(s)} > 0} \mathbb{P}_j^{(s)}(c), \min_{t \in [k_2] : \bar{\beta}^{(t)} > 0} \mathbb{Q}_j^{(t)}(c) \right\}.$$

The one-sided lower-bound selectors used by the two recursive sampling procedures are then defined by

$$\ell^{\mathbb{P}}(j, \bar{\alpha}, c) = \ell^{\mathbb{Q}}(j, \bar{\beta}, c) = \ell(j, \bar{\alpha}, \bar{\beta}, c).$$

By construction, $\ell^{\mathbb{P}}(j, \bar{\alpha}, c)$ satisfies (4) for $\mathbb{P}$, and $\ell^{\mathbb{Q}}(j, \bar{\beta}, c)$ satisfies (7) for $\mathbb{Q}$. In Line 4 of Algorithm 1, each process draws a random pair $(c, e) \in [q] \times \{0, 1\}$. Let $(c_{\bar{\mathbb{P}}}, e_{\bar{\mathbb{P}}})$ and $(c_{\bar{\mathbb{Q}}}, e_{\bar{\mathbb{Q}}})$ denote the pairs drawn by RecursiveSample$(j, \bar{\alpha})$ and RecursiveSample$(j, \bar{\beta})$, respectively. Since both procedures use the same lower bounds $\ell(j, \bar{\alpha}, \bar{\beta}, c)$, we define a coupling $\bar{\mathcal{C}}$ between these two pairs as follows:

- For each $c \in [q]$, $\mathbf{Pr}_{\bar{\mathcal{C}}}\left[c_{\bar{\mathbb{P}}} = c_{\bar{\mathbb{Q}}} = c \wedge e_{\bar{\mathbb{P}}} = e_{\bar{\mathbb{Q}}} = 0\right] = \ell(j, \bar{\alpha}, \bar{\beta}, c)$.
- For each $c \in [q]$, $\mathbf{Pr}_{\bar{\mathcal{C}}}\left[c_{\bar{\mathbb{P}}} = c_{\bar{\mathbb{Q}}} = c \wedge e_{\bar{\mathbb{P}}} = e_{\bar{\mathbb{Q}}} = 1\right] = \min\{\bar{\mathbb{P}}_j(c), \bar{\mathbb{Q}}_j(c)\} - \ell(j, \bar{\alpha}, \bar{\beta}, c)$, where $\bar{\mathbb{P}} = \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \bigotimes_{i=j}^{n} \mathbb{P}_i^{(s)}$ and $\bar{\mathbb{Q}} = \sum_{t=1}^{k_2} \bar{\beta}^{(t)} \bigotimes_{i=j}^{n} \mathbb{Q}_i^{(t)}$ are the mixtures of product distributions at the current recursive step.
- Given the above two points, for each $c \in A = \{x \in [q] \mid \bar{\mathbb{P}}_j(x) > \bar{\mathbb{Q}}_j(x)\}$, the random variable $(c_{\bar{\mathbb{P}}}, e_{\bar{\mathbb{P}}})$ has a remaining probability $\bar{\mathbb{P}}_j(c) - \bar{\mathbb{Q}}_j(c)$ of taking value $(c, 1)$; and for each $c \in B = \{x \in [q] \mid \bar{\mathbb{P}}_j(x) < \bar{\mathbb{Q}}_j(x)\}$, the random variable $(c_{\bar{\mathbb{Q}}}, e_{\bar{\mathbb{Q}}})$ has remaining probability $\bar{\mathbb{Q}}_j(c) - \bar{\mathbb{P}}_j(c)$ of taking value $(c, 1)$. The total remaining probabilities on the two sides are equal: $\sum_{c \in A}(\bar{\mathbb{P}}_j(c) - \bar{\mathbb{Q}}_j(c)) = \sum_{c \in B}(\bar{\mathbb{Q}}_j(c) - \bar{\mathbb{P}}_j(c))$. We therefore couple the remaining mass arbitrarily. Since A and B are disjoint, in this case we necessarily have

$$c_{\bar{\mathbb{P}}} \neq c_{\bar{\mathbb{Q}}} \quad \text{and} \quad e_{\bar{\mathbb{P}}} = e_{\bar{\mathbb{Q}}} = 1.$$

Using the coupling $\bar{\mathcal{C}}$ between $(c_{\bar{\mathbb{P}}}, e_{\bar{\mathbb{P}}})$ and $(c_{\bar{\mathbb{Q}}}, e_{\bar{\mathbb{Q}}})$, we now couple two sampling processes RecursiveSample$(j, \bar{\alpha})$ and RecursiveSample$(j, \bar{\beta})$ as follows:

- Sample $(c_{\bar{\mathbb{P}}}, e_{\bar{\mathbb{P}}})$ and $(c_{\bar{\mathbb{Q}}}, e_{\bar{\mathbb{Q}}})$ according to $\bar{\mathcal{C}}$.
- If $c_{\bar{\mathbb{P}}} = c_{\bar{\mathbb{Q}}} = c$ for some $c \in [q]$ and $e_{\bar{\mathbb{P}}} = e_{\bar{\mathbb{Q}}} = 0$, then we set $X_j = Y_j = c$ and recursively couple the suffixes via RecursiveCouple$(j+1, \bar{\alpha}, \bar{\beta})$.
- If $c_{\bar{\mathbb{P}}} = c_{\bar{\mathbb{Q}}} = c$ for some $c \in [q]$ and $e_{\bar{\mathbb{P}}} = e_{\bar{\mathbb{Q}}} = 1$, then we set $X_j = Y_j = c$ and recursively couple the suffixes via RecursiveCouple$(j+1, \bar{\alpha}_{j,c}, \bar{\beta}_{j,c})$.
- In the remaining case, we have $X_j = c_{\bar{\mathbb{P}}} \neq c_{\bar{\mathbb{Q}}} = Y_j$ and $e_{\bar{\mathbb{P}}} = e_{\bar{\mathbb{Q}}} = 1$. Hence the coupling already fails at coordinate j, and we simply sample the suffixes independently using RecursiveSample$(j+1, \bar{\alpha}_{j,c_{\bar{\mathbb{P}}}})$ and RecursiveSample$(j+1, \bar{\beta}_{j,c_{\bar{\mathbb{Q}}}})$.

The pseudocode of the coupling process is given in Algorithm 2.

▶ **Lemma 15** (Validity of recursive coupling). *Given two mixtures of product distributions $\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}$ and $\mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}_i^{(t)}$, fix a shared lower-bound selector ℓ as in Definition 14. Then RecursiveCouple$(1, \alpha, \beta)$ in Algorithm 2 outputs joint random variables (X, Y) such that $X \sim \mathbb{P}$ and $Y \sim \mathbb{Q}$.*

Lemma 15 follows by a straightforward induction. The proof can be found in the full version.

4.1.3 Discrepancy of recursive coupling

We now analyze the discrepancy of the recursive coupling process. Let $\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}$ and $\mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}_i^{(t)}$ be two mixtures of product distributions. Define

$$(X, Y) \leftarrow \text{RecursiveCouple}(1, \alpha, \beta).$$

Let $\mathcal{C}_{\mathrm{RC}}$ denote the joint distribution of (X, Y). By the coupling inequality, the discrepancy $\mathbf{Pr}_{\mathcal{C}_{\mathrm{RC}}}[X \neq Y]$ is always at least $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q})$. The following lemma provides an upper bound on the discrepancy of our recursive coupling.

▶ **Lemma 16** (Discrepancy of recursive coupling). *Let $[q]$ be a finite domain. For two mixtures of product distributions $\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}$ and $\mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}_i^{(t)}$ over $[q]^n$, the recursive coupling satisfies*

$$\mathbf{Pr}_{\mathcal{C}_{\mathrm{RC}}}[X \neq Y] \leq (4nq)^{k_1+k_2-1} \cdot d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q}).$$

The proof of Lemma 16 follows the known techniques of [20] and can be found in the full version.

Algorithm 2 Recursive coupling process for mixtures of product distributions.

RecursiveCouple $(j, \bar{\alpha}, \bar{\beta})$;
Parameter : an index $j \in [n+1]$ and mixing weights $\bar{\alpha} = (\bar{\alpha}^{(s)})_{s=1}^{k_1}$, $\bar{\beta} = (\bar{\beta}^{(t)})_{t=1}^{k_2}$
Static Input : product distributions $\mathbb{P}^{(1)}, \mathbb{P}^{(2)}, \ldots, \mathbb{P}^{(k_1)}$ and $\mathbb{Q}^{(1)}, \mathbb{Q}^{(2)}, \ldots, \mathbb{Q}^{(k_2)}$ over $[q]^n$; a fixed shared probability lower-bound selector ℓ as in Definition 14
Output : joint random variables (X, Y) such that $X = (X_j, X_{j+1}, \ldots, X_n) \sim \bar{\mathbb{P}} = \sum_{s=1}^{k_1} \bar{\alpha}^{(s)} \bigotimes_{i=j}^{n} \mathbb{P}_i^{(s)}$ and $Y = (Y_j, Y_{j+1}, \ldots, Y_n) \sim \bar{\mathbb{Q}} = \sum_{t=1}^{k_2} \bar{\beta}^{(t)} \bigotimes_{i=j}^{n} \mathbb{Q}_i^{(t)}$
if $j > n$ **then return** $(\emptyset, \emptyset)$;
Compute $\ell(j, \bar{\alpha}, \bar{\beta}, c)$ for all $c \in [q]$, and then use the coupling $\bar{\mathcal{C}}$ to jointly sample $(c_{\bar{\mathbb{P}}}, e_{\bar{\mathbb{P}}})$ and $(c_{\bar{\mathbb{Q}}}, e_{\bar{\mathbb{Q}}})$;
if $e_{\bar{\mathbb{P}}} = e_{\bar{\mathbb{Q}}} = 0$ **then**
    Sample $(X_{j+1:n}, Y_{j+1:n}) \leftarrow$ RecursiveCouple$(j+1, \bar{\alpha}, \bar{\beta})$;
else
    Let $\bar{\alpha}_{j,c_{\bar{\mathbb{P}}}}^{(s)} = \bar{\alpha}^{(s)} \frac{\mathbb{P}_j^{(s)}(c_{\bar{\mathbb{P}}}) - \ell(j, \bar{\alpha}, \bar{\beta}, c_{\bar{\mathbb{P}}})}{\bar{\mathbb{P}}_j(c_{\bar{\mathbb{P}}}) - \ell(j, \bar{\alpha}, \bar{\beta}, c_{\bar{\mathbb{P}}})}$ for $s \in [k_1]$ and $\bar{\beta}_{j,c_{\bar{\mathbb{Q}}}}^{(t)} = \bar{\beta}^{(t)} \frac{\mathbb{Q}_j^{(t)}(c_{\bar{\mathbb{Q}}}) - \ell(j, \bar{\alpha}, \bar{\beta}, c_{\bar{\mathbb{Q}}})}{\bar{\mathbb{Q}}_j(c_{\bar{\mathbb{Q}}}) - \ell(j, \bar{\alpha}, \bar{\beta}, c_{\bar{\mathbb{Q}}})}$ for $t \in [k_2]$;
    if $c_{\bar{\mathbb{P}}} = c_{\bar{\mathbb{Q}}}$ **then**
        Sample $(X_{j+1:n}, Y_{j+1:n}) \leftarrow$ RecursiveCouple$(j+1, \bar{\alpha}_{j,c}, \bar{\beta}_{j,c})$, where $c = c_{\bar{\mathbb{P}}} = c_{\bar{\mathbb{Q}}}$;
    else
        Sample $X_{j+1:n} \leftarrow$ RecursiveSample$(j+1, \bar{\alpha}_{j,c_{\bar{\mathbb{P}}}})$ with $\ell^{\mathbb{P}} = 0$;
        Sample $Y_{j+1:n} \leftarrow$ RecursiveSample$(j+1, \bar{\beta}_{j,c_{\bar{\mathbb{Q}}}})$ with $\ell^{\mathbb{Q}} = 0$;
Let $X = (X_j = c_{\bar{\mathbb{P}}}, X_{j+1:n})$ and $Y = (Y_j = c_{\bar{\mathbb{Q}}}, Y_{j+1:n})$ and return (X, Y);

4.2 Implementation of oracles

In this section, we implement the oracle queries in Assumption 9 based on Algorithm 2. Consider the execution of Algorithm 2 on two mixtures of product distributions $\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}_i^{(s)}$ and $\mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}_i^{(t)}$. Define the following set of states and transitions between them.

For each recursion instance RecursiveCouple$(j, \bar{\alpha}, \bar{\beta})$, we create a state S with label $(j, \bar{\alpha}, \bar{\beta})$, where $1 \le j \le n+1$. A state S is said to be in layer j if j is the first coordinate of the label. We add a special failure state $S = \perp$ to denote that the coupling fails. Let $\mathcal{S}$ denote the set of all possible states that can be reached by the coupling process with a *positive* probability, including the failure state $\perp$.

For each state $S \in \mathcal{S}$ with label $(j, \bar{\alpha}, \bar{\beta})$ for $j \in [n]$, we define the following transitions according to the coupling process. The coupling process samples $(c_{\bar{\mathbb{P}}}, e_{\bar{\mathbb{P}}})$ and $(c_{\bar{\mathbb{Q}}}, e_{\bar{\mathbb{Q}}})$ for $\bar{\mathbb{P}}$ and $\bar{\mathbb{Q}}$, respectively, and then makes the transition.

- **Type-I**: For any $c \in [q]$ with $\ell(j, \bar{\alpha}, \bar{\beta}, c) > 0$, with probability $\ell(j, \bar{\alpha}, \bar{\beta}, c)$, S moves to a new state S' with label $(j+1, \bar{\alpha}, \bar{\beta})$. Define a transition t from S to S' with weight $w(t) = \ell(j, \bar{\alpha}, \bar{\beta}, c)$ and label $L(t) = (c, c)$. This means that S moves to S' through t with probability $\ell(j, \bar{\alpha}, \bar{\beta}, c)$ and sets $X_j = Y_j = c$.
- **Type-II**: For any $c \in [q]$ with $p_c \triangleq \min\{\bar{\mathbb{P}}_j(c), \bar{\mathbb{Q}}_j(c)\} - \ell(j, \bar{\alpha}, \bar{\beta}, c) > 0$, with probability p_c, S moves to a new state S' with label $(j+1, \bar{\alpha}_{j,c}, \bar{\beta}_{j,c})$. Define a transition t from S to S' with weight $w(t) = p_c$ and label $L(t) = (c, c)$. This means that S moves to S' through t with probability p_c and sets $X_j = Y_j = c$.

- **Type-III**: With the remaining probability, the coupling fails ($X_j \neq Y_j$). For any $c, c' \in [q]$ with $c \neq c'$ and $p_{c,c'} \triangleq \mathbf{Pr}\left[c_{\bar{\mathbb{P}}} = c \wedge c_{\bar{\mathbb{Q}}} = c'\right] > 0$, define a transition t from S to the failure state $\perp$ with weight $w(t) = p_{c,c'}$ and label $L(t) = (X_{j:n}, Y_{j:n})$, where $X_j = c$, $Y_j = c'$, and $X_{j+1:n}, Y_{j+1:n}$ are sampled independently in Lines 11 and 12 of Algorithm 2. Hence, the label contains a pair of random configurations, both of length $n - j + 1$. The label $L(t)$ is random because we further sample the pair $(X_{j+1:n}, Y_{j+1:n})$. We remark that there are multiple transitions from S to the failure state $\perp$ with different weights and labels.

▶ **Lemma 17.** *The total number of states is bounded by* $|\mathcal{S}| \leq (nq+1)^{k_1+k_2-1} + 1$.

The proof can be found in the full version.

The oracle queries in Assumption 9 can be implemented by the following algorithm. One execution of the coupling process can be represented as a random walk on the state space $\mathcal{S}$. The random walk starts from the root state $S = S_{\text{root}} = (1, \alpha, \beta)$. At each step, the random walk samples a transition $t = (S \to S')$ from S with probability $w(t)$ and moves to the next state S'. The random walk ends either at a state in layer $n+1$ (coupling succeeds) or at the failure state $\perp$ (coupling fails). The coupling process outputs a pair $(X_{1:n}, Y_{1:n})$, which is determined by the labels of all transitions along the random walk.

Preprocessing step. We build the DAG with all states in $\mathcal{S}$ and all transitions between them. Using Lemma 17, the total number of states and transitions in the DAG is at most $O(q^2(nq+1)^{k_1+k_2-1})$. For each state $S \in \mathcal{S}$, compute $p_{\text{fail}}(S)$, the probability that a random walk starting from S reaches the failure state $\perp$. By definition,

$$p_{\text{fail}}(\perp) = 1 \text{ and } p_{\text{fail}}(S) = 0 \text{ for all } S \text{ in layer } n+1.$$

For each non-failed state $S \in \mathcal{S}$ in layer $j \in [n]$, we have the following recursive relation:

$$p_{\text{fail}}(S) = \sum_{t=(S\to S') \in \text{transitions from } S} w(t) p_{\text{fail}}(S').$$

Using dynamic programming, we can compute $p_{\text{fail}}(S)$ for all $S \in \mathcal{S}$ in time $O(q^2(nq+1)^{k_1+k_2-1})$. By the definition of $p_{\text{fail}}(S)$, the failure probability of the recursive coupling is

$$\mathbf{Pr}_{\mathcal{C}_{\text{RC}}}[X \neq Y] = p_{\text{fail}}(S_{\text{root}}).$$

Sampling query. Next, we give an algorithm that generates a random sample of X conditional on $X \neq Y$ in the recursive coupling. Equivalently, we can sample a random walk on the DAG starting from the root state S_{root}, conditional on the random walk reaching the failure state $\perp$. If $p_{\text{fail}}(S_{\text{root}}) = 0$, then $d_{\text{TV}}(\mathbb{P}, \mathbb{Q}) = 0$ by the coupling inequality, and this sampling query is not needed. Otherwise, the query can be implemented by the following algorithm.

- Initialize $S = S_{\text{root}}$.
- Sample a random transition $t = (S \to S')$ from S with probability $\frac{w(t)p_{\text{fail}}(S')}{p_{\text{fail}}(S)}$ and then update the current state S to S'.
- The process terminates when S reaches the failure state $\perp$.

The above process gives a random sequence of transitions on the DAG. By concatenating the labels in the sequence, we obtain a random pair $(X, Y) \in [q]^n \times [q]^n$, and we output the sample $X \in [q]^n$.

The correctness of the conditional sample X follows directly from the definition of $p_{\text{fail}}(\cdot)$. The running time to generate a random sample X is $O(q^2(nq+1)^{k_1+k_2-1})$.

Evaluation query. Given a fixed configuration $\sigma \in [q]^n$, we want to evaluate the probability

$$\mathbf{Pr}_{\mathcal{C}_{\mathrm{RC}}}[X = \sigma \wedge X \neq Y].$$

For each non-failed state $S \in \mathcal{S}$ in layer $j \in [n]$, let $p^{\sigma}_{\mathrm{fail}}(S)$ be the probability that a random walk starting from S reaches the failure state $\perp$ and that the label $X_{j:n}$ of the random walk is exactly $\sigma_{j:n}$. By definition, since S_{root} is the layer-1 state, it holds that

$$p^{\sigma}_{\mathrm{fail}}(S_{\mathrm{root}}) = \mathbf{Pr}_{\mathcal{C}_{\mathrm{RC}}}[X = \sigma \wedge X \neq Y].$$

For each $S \in \mathcal{S}$ in layer $j = n+1$, by definition, we have $p^{\sigma}_{\mathrm{fail}}(S) = 0$.
Fix any non-failed state $S \in \mathcal{S}$ in layer $j \in [n]$ with $S = (j, \bar{\alpha}, \bar{\beta})$. Define

$$\mathcal{T} = \{t \mid t \text{ is a transition from } S \text{ to } S' \neq \perp \text{ and } L(t) = (\sigma_j, \sigma_j)\}.$$

The above definition considers all transitions from S to non-failed states S'. We still need to consider the type-III transitions from S to the failure state $\perp$. By definition, there are at most $q(q-1)$ such transitions. Let $\mathcal{T}_{\perp}(S)$ denote the set of all type-III transitions from S to the failure state $\perp$. Every such transition can be specified by a pair (c, c') with $c \neq c'$. We use $t_{c,c'}$ to denote the corresponding type-III transition. Note that the weight $w(t_{c,c'})$ is $\mathbf{Pr}\left[c_{\bar{\mathbb{P}}} = c \wedge c_{\bar{\mathbb{Q}}} = c'\right]$ in the coupling at Line 3 of Algorithm 2. The label of $t_{c,c'}$ is a random pair $(X_{j:n}, Y_{j:n})$, where $X_j = c$, $Y_j = c'$, and $X_{j+1:n}$ and $Y_{j+1:n}$ are sampled independently in Lines 11 and 12 of Algorithm 2. We define

$$\tau(t_{c,c'}, \sigma) \triangleq \mathbf{Pr}[X_{j+1:n} = \sigma_{j+1:n}] = \sum_{s=1}^{k_1} \bar{\alpha}^{(s)}_{j,c} \prod_{i=j+1}^{n} \mathbb{P}^{(s)}_i(\sigma_i).$$

Now, we have the following recursive relation:

$$p^{\sigma}_{\mathrm{fail}}(S) = \sum_{t=(S \to S') \in \mathcal{T}} w(t) p^{\sigma}_{\mathrm{fail}}(S') + \sum_{t_{c,c'} \in \mathcal{T}_{\perp}(S): c = \sigma_j} w(t_{c,c'}) \tau(t_{c,c'}, \sigma).$$

Again, we can use dynamic programming on the DAG from layer $n+1$ to layer 1 to compute all values $p^{\sigma}_{\mathrm{fail}}(S)$ for all $S \in \mathcal{S}$ in time $O(q^2(nq+1)^{k_1+k_2-1})$. The final result is $p^{\sigma}_{\mathrm{fail}}(S_{\mathrm{root}})$.

5 Algorithm for Mixtures of Boolean Subcubes

5.1 Reduction to a counting problem

Consider two mixtures of Boolean subcubes $\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^{n} \mathbb{P}^{(s)}_i$ and $\mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^{n} \mathbb{Q}^{(t)}_i$. A component $\mathbb{P}^{(s)}$ over $\{0,1\}^n$ is a Boolean subcube if, for any $i \in [n]$, $\mathbb{P}^{(s)}_i(1) \in \{0, \frac{1}{2}, 1\}$. Thus, for any $\omega \in \{0,1\}^n$, $\mathbb{P}^{(s)}(\omega) = \prod_{i=1}^{n} \mathbb{P}^{(s)}_i(\omega_i)$ is a product of values in $\{0, \frac{1}{2}, 1\}$. We define the following partition $\Lambda^{\mathrm{one}}_{\mathbb{P}^{(s)}}$, $\Lambda^{\mathrm{zero}}_{\mathbb{P}^{(s)}}$, and $\Lambda^{\mathrm{half}}_{\mathbb{P}^{(s)}}$ of $[n]$:

$$\Lambda^{\mathrm{one}}_{\mathbb{P}^{(s)}} = \{i \in [n] \mid \mathbb{P}^{(s)}_i(1) = 1\},\ \Lambda^{\mathrm{zero}}_{\mathbb{P}^{(s)}} = \{i \in [n] \mid \mathbb{P}^{(s)}_i(1) = 0\},$$
$$\Lambda^{\mathrm{half}}_{\mathbb{P}^{(s)}} = \left\{i \in [n] \mid \mathbb{P}^{(s)}_i(1) = \frac{1}{2}\right\}.$$

In words, for the product distribution $\mathbb{P}^{(s)}$, the set $\Lambda^{\mathrm{one}}_{\mathbb{P}^{(s)}}$ contains all dimensions whose value is fixed to 1, the set $\Lambda^{\mathrm{zero}}_{\mathbb{P}^{(s)}}$ contains all dimensions whose value is fixed to 0, and the set $\Lambda^{\mathrm{half}}_{\mathbb{P}^{(s)}}$ contains all dimensions whose value is sampled uniformly at random. Similarly, for a component $\mathbb{Q}^{(t)}$, we define the sets $\Lambda^{\mathrm{one}}_{\mathbb{Q}^{(t)}}$, $\Lambda^{\mathrm{zero}}_{\mathbb{Q}^{(t)}}$, and $\Lambda^{\mathrm{half}}_{\mathbb{Q}^{(t)}}$ in the same way.

The probability of each $\omega \in \{0,1\}^n$ can be written as follows:

$$\mathbb{P}^{(s)}(\omega) = \left(\frac{1}{2}\right)^{\left|\Lambda^{\text{half}}_{\mathbb{P}^{(s)}}\right|} \cdot \mathbb{1}\left[\forall i \in \Lambda^{\text{one}}_{\mathbb{P}^{(s)}}, \omega_i = 1\right] \cdot \mathbb{1}\left[\forall i \in \Lambda^{\text{zero}}_{\mathbb{P}^{(s)}}, \omega_i = 0\right],$$

$$\mathbb{Q}^{(t)}(\omega) = \left(\frac{1}{2}\right)^{\left|\Lambda^{\text{half}}_{\mathbb{Q}^{(t)}}\right|} \cdot \mathbb{1}\left[\forall i \in \Lambda^{\text{one}}_{\mathbb{Q}^{(t)}}, \omega_i = 1\right] \cdot \mathbb{1}\left[\forall i \in \Lambda^{\text{zero}}_{\mathbb{Q}^{(t)}}, \omega_i = 0\right],$$

where the indicator function $\mathbb{1}[*]$ returns 1 if statement $*$ holds and 0 otherwise. For the distribution $\mathbb{P}^{(s)}$, we say $\omega \in \{0,1\}^n$ is *feasible* if for all $i \in \Lambda^{\text{one}}_{\mathbb{P}^{(s)}}$, $\omega_i = 1$ and for all $i \in \Lambda^{\text{zero}}_{\mathbb{P}^{(s)}}$, $\omega_i = 0$. Each feasible $\omega \in \{0,1\}^n$ has probability $(1/2)^{|\Lambda^{\text{half}}_{\mathbb{P}^{(s)}}|}$ under the distribution $\mathbb{P}^{(s)}$, while each infeasible $\omega \in \{0,1\}^n$ has probability 0. A similar structural property holds for the distribution $\mathbb{Q}^{(t)}$.

The feasibility of a configuration $\omega \in \{0,1\}^n$ under the distribution $\mathbb{P}^{(s)}$ can be expressed by a Boolean formula $F_{\mathbb{P}^{(s)}} : \{0,1\}^n \to \{0,1\}$ as follows:

$$F_{\mathbb{P}^{(s)}}(\omega) = \bigwedge_{i \in \Lambda^{\text{one}}_{\mathbb{P}^{(s)}}} \omega_i \wedge \bigwedge_{i \in \Lambda^{\text{zero}}_{\mathbb{P}^{(s)}}} \neg\omega_i. \tag{8}$$

Similarly, for the distribution $\mathbb{Q}^{(t)}$, define the Boolean formula $F_{\mathbb{Q}^{(t)}} : \{0,1\}^n \to \{0,1\}$ as follows:

$$F_{\mathbb{Q}^{(t)}}(\omega) = \bigwedge_{i \in \Lambda^{\text{one}}_{\mathbb{Q}^{(t)}}} \omega_i \wedge \bigwedge_{i \in \Lambda^{\text{zero}}_{\mathbb{Q}^{(t)}}} \neg\omega_i. \tag{9}$$

By definition, $\mathbb{P}^{(s)}(\omega) = (1/2)^{|\Lambda^{\text{half}}_{\mathbb{P}^{(s)}}|} \cdot F_{\mathbb{P}^{(s)}}(\omega)$ and $\mathbb{Q}^{(t)}(\omega) = (1/2)^{|\Lambda^{\text{half}}_{\mathbb{Q}^{(t)}}|} \cdot F_{\mathbb{Q}^{(t)}}(\omega)$. The total variation distance between mixtures of Boolean subcubes $\mathbb{P} = \sum_{s=1}^{k_1} \alpha^{(s)} \bigotimes_{i=1}^n \mathbb{P}_i^{(s)}$ and $\mathbb{Q} = \sum_{t=1}^{k_2} \beta^{(t)} \bigotimes_{i=1}^n \mathbb{Q}_i^{(t)}$ can be expressed as follows:

$$\begin{aligned}
d_{\text{TV}}(\mathbb{P}, \mathbb{Q}) &= \frac{1}{2} \sum_{\omega \in \{0,1\}^n} |\mathbb{P}(\omega) - \mathbb{Q}(\omega)| \\
&= \frac{1}{2} \sum_{\omega \in \{0,1\}^n} \left| \sum_{s=1}^{k_1} \alpha^{(s)} \prod_{i=1}^n \mathbb{P}_i^{(s)}(\omega_i) - \sum_{t=1}^{k_2} \beta^{(t)} \prod_{i=1}^n \mathbb{Q}_i^{(t)}(\omega_i) \right| \\
&= \frac{1}{2} \sum_{\omega \in \{0,1\}^n} \left| \sum_{s=1}^{k_1} \alpha^{(s)} \cdot \left(\frac{1}{2}\right)^{\left|\Lambda^{\text{half}}_{\mathbb{P}^{(s)}}\right|} \cdot F_{\mathbb{P}^{(s)}}(\omega) - \sum_{t=1}^{k_2} \beta^{(t)} \cdot \left(\frac{1}{2}\right)^{\left|\Lambda^{\text{half}}_{\mathbb{Q}^{(t)}}\right|} \cdot F_{\mathbb{Q}^{(t)}}(\omega) \right|.
\end{aligned}$$

Define the *characteristic function* $F : \{0,1\}^n \to \{0,1\}^{k_1+k_2}$ as follows:

$$F(\omega) = \left(F_{\mathbb{P}^{(1)}}(\omega), F_{\mathbb{P}^{(2)}}(\omega), \ldots, F_{\mathbb{P}^{(k_1)}}(\omega), F_{\mathbb{Q}^{(1)}}(\omega), F_{\mathbb{Q}^{(2)}}(\omega), \ldots, F_{\mathbb{Q}^{(k_2)}}(\omega)\right).$$

By the above calculation, the value of $|\mathbb{P}(\omega) - \mathbb{Q}(\omega)|$ is fully determined by $F(\omega)$. The key observation is that the image space of F has constant size $2^{k_1+k_2}$, so we can reorganize all $\omega \in \{0,1\}^n$ according to the value of the characteristic function $F(\omega)$. Formally,

$$d_{\text{TV}}(\mathbb{P}, \mathbb{Q}) = \frac{1}{2} \sum_{\chi \in \{0,1\}^{k_1+k_2}} N_\chi \cdot \left| \sum_{s=1}^{k_1} \alpha^{(s)} \cdot \left(\frac{1}{2}\right)^{\left|\Lambda^{\text{half}}_{\mathbb{P}^{(s)}}\right|} \cdot \chi_s - \sum_{t=1}^{k_2} \beta^{(t)} \cdot \left(\frac{1}{2}\right)^{\left|\Lambda^{\text{half}}_{\mathbb{Q}^{(t)}}\right|} \cdot \chi_{t+k_1} \right| \tag{10}$$

where N_χ counts the number of $\omega \in \{0,1\}^n$ such that $F(\omega) = \chi$, i.e.,

$$N_\chi = |\{\omega \in \{0,1\}^n \mid F(\omega) = \chi\}|.$$

By (10), computing the total variation distance between two mixtures of Boolean subcubes $\mathbb{P}$ and $\mathbb{Q}$ reduces to counting N_χ for all $\chi \in \{0,1\}^{k_1+k_2}$.

5.2 The counting algorithm

▶ **Lemma 18.** *Given any $\chi \in \{0,1\}^{k_1+k_2}$, the value of N_χ can be computed in time $O(2^{|S_0|} \cdot n(k_1+k_2))$, where $S_0 = \{\, j \in [k_1+k_2] \mid \chi_j = 0 \,\}$ is the set of indices j with $\chi_j = 0$.*

Fix $\chi \in \{0,1\}^{k_1+k_2}$, and consider the set of $\omega \in \{0,1\}^n$ such that $F(\omega) = \chi$. We need to count the number of such $\omega \in \{0,1\}^n$. To simplify the notation, define a sequence of Boolean functions $F_1, F_2, \ldots, F_{k_1+k_2} : \{0,1\}^n \to \{0,1\}$ as follows:

$$\forall j \in [k_1+k_2], \quad F_j(\omega) = \begin{cases} F_{\mathbb{P}^{(j)}}(\omega) & \text{if } j \in [k_1], \\ F_{\mathbb{Q}^{(j-k_1)}}(\omega) & \text{if } j \in [k_1+1, k_1+k_2], \end{cases}$$

where $F_{\mathbb{P}^{(s)}}$ and $F_{\mathbb{Q}^{(t)}}$ are defined in (8) and (9), respectively. For a configuration $\omega \in \{0,1\}^n$ with $F(\omega) = \chi$, we have $F_j(\omega) = \chi_j$ for all $j \in [k_1+k_2]$.

For a subset $S \subseteq [k_1+k_2]$ of indices, define the following set of configurations:

$$\Phi(S) = \{\omega \in \{0,1\}^n \mid F_j(\omega) = 1 \text{ for all } j \in S\}.$$

Let $S_1 = \{j \in [k_1+k_2] \mid \chi_j = 1\}$ and $S_0 = \{j \in [k_1+k_2] \mid \chi_j = 0\}$. By the definition of N_χ and the inclusion-exclusion principle, we have

$$N_\chi = |\Phi(S_1)| - |\{\omega \in \Phi(S_1) \mid \exists\, j \in S_0, F_j(\omega) = 1\}| = \sum_{S \subseteq S_0} (-1)^{|S|} \cdot |\Phi(S_1 \cup S)|. \tag{11}$$

▶ **Observation 19.** *Given any $S \subseteq [k_1+k_2]$, the size $|\Phi(S)|$ can be computed in time $O(n \cdot |S|)$.*

Proof. By the definition of $F_{\mathbb{P}^{(s)}}$ and $F_{\mathbb{Q}^{(t)}}$, we have $F_{\mathbb{P}^{(s)}}(\omega) = 1$ iff $\omega_i = 1$ for all $i \in \Lambda^{\text{one}}_{\mathbb{P}^{(s)}}$ and $\omega_i = 0$ for all $i \in \Lambda^{\text{zero}}_{\mathbb{P}^{(s)}}$. Similarly, $F_{\mathbb{Q}^{(t)}}(\omega) = 1$ iff $\omega_i = 1$ for all $i \in \Lambda^{\text{one}}_{\mathbb{Q}^{(t)}}$ and $\omega_i = 0$ for all $i \in \Lambda^{\text{zero}}_{\mathbb{Q}^{(t)}}$. We can go through all $j \in S$ and fix the values of the corresponding dimensions $i \in [n]$. If there is a contradiction, then $|\Phi(S)| = 0$. Otherwise, $|\Phi(S)| = 2^r$, where r is the number of dimensions whose values are not fixed. The running time is $O(n \cdot |S|)$. ◀

We now prove Lemma 18. Combining (11) and Observation 19, we can exactly compute the value of N_χ in time

$$\sum_{S \subseteq S_0} O(n|S_1 \cup S|) \le \sum_{0 \le k \le |S_0|} \binom{|S_0|}{k} \cdot O(n(k_1+k_2)) = O(2^{|S_0|} \cdot n(k_1+k_2)).$$

5.3 The total variation distance algorithm

The algorithm for computing the total variation distance between two mixtures of Boolean subcubes $\mathbb{P}$ and $\mathbb{Q}$ is obtained by combining (10) and Lemma 18. Let $K = k_1 + k_2$. The overall running time is

$$\sum_{0 \le m \le K} \binom{K}{m} \cdot O(2^m \cdot nK) = O(3^K K \cdot n) = O(3^{k_1+k_2}(k_1+k_2) \cdot n).$$

Here m enumerates the number of indices j with $\chi_j = 0$. This proves the main result in Theorem 4.

6 Proof of the hardness result

Proof of Theorem 5. We show that if there is a polynomial-time algorithm for computing the TV-distance between two mixtures of Boolean subcubes with $k_1 + k_2 = \Theta(n)$, then there is a polynomial-time algorithm for #3SAT, restricted to 3-CNF formulas. Let

$$\varphi(x_1, x_2, \ldots, x_r) = C_1 \wedge C_2 \wedge \ldots \wedge C_m$$

be a 3-CNF formula over the used variables $x_1, x_2, \ldots, x_r$, and $m \geq 1$. Set $N = \max\{r, m\}$. If $N > r$, we add $N - r$ dummy variables $x_{r+1}, \ldots, x_N$. Let S denote the number of satisfying assignments of φ over $x_1, \ldots, x_N$. As the dummy variables do not appear in any clause, the number of satisfying assignments for φ is given by

$$\#\mathsf{SAT}(\varphi) = \frac{S}{2^{N-r}}.$$

We establish the proof by showing the relation between S and the TV-distance between certain mixtures of Boolean subcubes. Let $n = N + 1$. We construct two mixtures of subcubes $\mathbb{P}$ and $\mathbb{Q}$ over $\{0,1\}^n$. A point in this space can be represented as $(b, \omega_1, \ldots, \omega_N)$, where $b \in \{0,1\}$ is an additional bit, and $(\omega_1, \ldots, \omega_N)$ is an assignment to $(x_1, \ldots, x_N)$. For each $j \in [m]$, let $x_{j,a}, x_{j,b}, x_{j,c}$ be the three variables appearing in the clause C_j. Define $(\omega_{j,a}, \omega_{j,b}, \omega_{j,c}) \in \{0,1\}^3$ to be the unique assignment to $x_{j,a}, x_{j,b}, x_{j,c}$ that falsifies C_j. Let U_j be a subcube such that

$$\mathbf{Pr}_{U_j}\left[(b, x_{j,a}, x_{j,b}, x_{j,c}) = (0, \omega_{j,a}, \omega_{j,b}, \omega_{j,c})\right] = 1,$$

and the other $N - 3$ coordinates are uniform over $\{0,1\}$. Define

$$\mathbb{P} = \frac{1}{m}\sum_{j=1}^{m} U_j,$$

is a mixture of $k_1 = m$ subcubes. Next we define $\mathbb{Q}$. Let V_0 and V_1 be two subcubes such that

$$\mathbf{Pr}_{V_0}[b = 0] = \mathbf{Pr}_{V_1}[b = 1] = 1,$$

and that are uniform over $(x_1, \ldots, x_N)$. Let $\lambda = \frac{1}{2m}$, and define

$$\mathbb{Q} = \lambda V_0 + (1 - \lambda) V_1,$$

which is a mixture of $k_2 = 2$ subcubes.

We now compute $d_{\mathrm{TV}}(\mathbb{P}, \mathbb{Q})$. For any assignment $\omega = (\omega_1, \ldots, \omega_N)$, let

$$F(\omega) = |\{j \in [m] \mid \omega \text{ falsifies } C_j\}|$$

denote the number of clauses falsified by ω. For every $\omega \in \{0,1\}^N$, we have

$$\mathbb{P}(0,\omega) = \frac{1}{m}\sum_{j=1}^{m} U_j(0,\omega) = \frac{1}{m}\sum_{j=1}^{m} \mathbf{1}[\omega \text{ falsifies } C_j]\cdot 2^{-(N-3)} = \frac{F(\omega)}{m}\cdot 2^{-(N-3)} = \frac{8F(\omega)}{m}\cdot 2^{-N},$$

and $\mathbb{P}(1,\omega) = 0$ by definition of U_j. Remark that

$$\mathbb{Q}(0,\omega) = \lambda \cdot 2^{-N} = \frac{1}{2m}\cdot 2^{-N} < \frac{8F(\omega)}{m}\cdot 2^{-N} = \mathbb{P}(0,\omega) \quad \text{iff } F(\omega) \geq 1;$$

$$\mathbb{Q}(1,\omega) = (1-\lambda)\cdot 2^{-N} > 0 = \mathbb{P}(1,\omega).$$

Therefore

$$\begin{aligned} d_{\mathrm{TV}}(\mathbb{P},\mathbb{Q}) &= \sum_{(b,\omega)\in\{0,1\}^{N+1}} \max\{0, \mathbb{P}(b,\omega) - \mathbb{Q}(b,\omega)\} = \sum_{\omega\in\{0,1\}^N : F(\omega)\geq 1} \mathbb{P}(0,\omega) - \mathbb{Q}(0,\omega) \\ &= \sum_{\omega\in\{0,1\}^N : F(\omega)\geq 1} \frac{8F(\omega)}{m}\cdot 2^{-N} - \sum_{\omega\in\{0,1\}^N : F(\omega)\geq 1} \frac{1}{2m}\cdot 2^{-N}. \end{aligned} \tag{12}$$

For the first term, we consider counting all pairs (ω, C_j) such that ω falsifies C_j. On the one hand, for each ω, it falsifies $F(\omega)$ clauses; on the other hand, for each C_j, it is falsified by 2^{N-3} assignments. Thus

$$\sum_{\omega\in\{0,1\}^N : F(\omega)\geq 1} F(\omega) = \sum_{\omega\in\{0,1\}^N} F(\omega) = m\cdot 2^{N-3}.$$

For the second term, there are S assignments ω satisfying all clauses, that is, $F(\omega) = 0$. So there are $2^N - S$ assignments satisfying $F(\omega) \geq 1$. Substituting back to (12), we have

$$\begin{aligned} d_{\mathrm{TV}}(\mathbb{P},\mathbb{Q}) &= \sum_{\omega\in\{0,1\}^N : F(\omega)\geq 1} \frac{8F(\omega)}{m}\cdot 2^{-N} - \sum_{\omega\in\{0,1\}^N : F(\omega)\geq 1} \frac{1}{2m}\cdot 2^{-N} \\ &= \frac{2^{-N+3}}{m}\cdot(m\cdot 2^{N-3}) - \frac{2^{-N}}{2m}\cdot(2^N - S) = 1 - \frac{1}{2m} + \frac{2^{-N}}{2m}S. \end{aligned}$$

Thus the number of satisfying assignments is given by

$$\#\mathsf{SAT}(\varphi) = \frac{S}{2^{N-r}} = 2^r(2m\cdot d_{\mathrm{TV}}(\mathbb{P},\mathbb{Q}) - 2m + 1).$$

The constructed distributions live on the $n = N+1$ dimensional subcube, with $k_1 + k_2 = m + 2$ total components. Since $N = \max\{r, m\}$ and $r \leq 3m$, we have $m \leq N \leq 3m$, hence

$$k_1 + k_2 = m + 2 = \Theta(N+1) = \Theta(n).$$

Therefore, an algorithm for Problem 3 with $k_1 + k_2 = \Theta(n)$ yields an exact algorithm for #3SAT. ◀

References

1 Hassan Ashtiani, Shai Ben-David, Nicholas J. A. Harvey, Christopher Liaw, Abbas Mehrabian, and Yaniv Plan. Nearly tight sample complexity bounds for learning mixtures of Gaussians via sample compression schemes. In *NeurIPS*, pages 3416–3425, 2018. URL: `https://proceedings.neurips.cc/paper/2018/hash/70ece1e1e0931919438fcfc6bd5f199c-Abstract.html`.

2 Arnab Bhattacharyya, Sutanu Gayen, Kuldeep S. Meel, Dimitrios Myrisiotis, A. Pavan, and N. V. Vinodchandran. On approximating total variation distance. In *IJCAI*, pages 3479–3487. ijcai.org, 2023. `doi:10.24963/IJCAI.2023/387`.

3 Arnab Bhattacharyya, Sutanu Gayen, Kuldeep S. Meel, Dimitrios Myrisiotis, A. Pavan, and N. V. Vinodchandran. Total variation distance meets probabilistic inference. In *ICML*. OpenReview.net, 2024.

4 Arnab Bhattacharyya, Sutanu Gayen, Kuldeep S. Meel, Dimitrios Myrisiotis, Aduri Pavan, and N. V. Vinodchandran. Computational explorations of total variation distance. In *ICLR*. OpenReview.net, 2025. URL: `https://openreview.net/forum?id=xak8c9l1nu`.

5 Arnab Bhattacharyya, Sutanu Gayen, Kuldeep S. Meel, and N. V. Vinodchandran. Efficient distance approximation for structured high-dimensional distributions via learning. In *NeurIPS*, 2020.

6 Guy Blanc, Jane Lange, Ali Malik, and Li-Yang Tan. Lifting uniform learners via distributional decomposition. In *STOC*, pages 1755–1767. ACM, 2023. doi:10.1145/3564246.3585212.

7 Guy Blanc, Jane Lange, Carmen Strassle, and Li-Yang Tan. A distributional-lifting theorem for PAC learning. In Nika Haghtalab and Ankur Moitra, editors, *COLT*, Proceedings of Machine Learning Research, pages 375–379. PMLR, 2025. URL: https://proceedings.mlr.press/v291/blanc25a.html.

8 Clément L. Canonne and Ronitt Rubinfeld. Testing probability distributions underlying aggregated data. In *ICALP*, volume 8572 of *Lecture Notes in Computer Science*, pages 283–295. Springer, 2014. doi:10.1007/978-3-662-43948-7_24.

9 Sitan Chen and Ankur Moitra. Beyond the low-degree algorithm: mixtures of subcubes and their applications. In *STOC*, pages 869–880. ACM, 2019. doi:10.1145/3313276.3316375.

10 Taolue Chen and Stefan Kiefer. On the total variation distance of labelled markov chains. In *LICS*, pages 33:1–33:10. ACM, 2014. doi:10.1145/2603088.2603099.

11 Jon Feldman, Ryan O'Donnell, and Rocco A. Servedio. Learning mixtures of product distributions over discrete domains. *SIAM*, 37(5):1536–1564, 2008. doi:10.1137/060670705.

12 Weiming Feng, Heng Guo, Mark Jerrum, and Jiaheng Wang. A simple polynomial-time approximation algorithm for the total variation distance between two product distributions. *TheoretiCS*, 2, 2023. doi:10.46298/THEORETICS.23.7.

13 Weiming Feng, Hongyang Liu, and Minji Yang. Approximating the total variation distance between spin systems. In *COLT*, volume 291 of *Proceedings of Machine Learning Research*, pages 1974–2025. PMLR, 2025. URL: https://proceedings.mlr.press/v291/feng25a.html.

14 Weiming Feng, Liqiang Liu, and Tianren Liu. On deterministically approximating total variation distance. In *SODA*, pages 1766–1791. SIAM, 2024. doi:10.1137/1.9781611977912.70.

15 Spencer Gordon, Bijan H. Mazaheri, Yuval Rabani, and Leonard J. Schulman. Source identification for mixtures of product distributions. In *COLT*, Proceedings of Machine Learning Research, pages 2193–2216. PMLR, 2021. URL: http://proceedings.mlr.press/v134/gordon21a.html.

16 Spencer L. Gordon, Erik Jahn, Bijan Mazaheri, Yuval Rabani, and Leonard J. Schulman. Identification of mixtures of discrete product distributions in near-optimal sample and time complexity. In *COLT*, Proceedings of Machine Learning Research, pages 2071–2091. PMLR, 2024. URL: https://proceedings.mlr.press/v247/gordon24a.html.

17 Prateek Jain and Sewoong Oh. Learning mixtures of discrete product distributions using spectral decompositions. In *COLT*, JMLR Workshop and Conference Proceedings, pages 824–856. JMLR.org, 2014. URL: http://proceedings.mlr.press/v35/jain14.html.

18 Chinmaya Kausik, Kevin Tan, and Ambuj Tewari. Learning mixtures of Markov Chains and MDPs. In *ICML*, Proceedings of Machine Learning Research, pages 15970–16017. PMLR, 2023. URL: https://proceedings.mlr.press/v202/kausik23a.html.

19 Stefan Kiefer. On computing the total variation distance of hidden Markov models. In *ICALP*, volume 107 of *LIPIcs*, pages 130:1–130:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.130.

20 Jeongyeol Kwon, Yonathan Efroni, Constantine Caramanis, and Shie Mannor. Reward-Mixing MDPs with few latent contexts are learnable. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 18057–18082. PMLR, 2023. URL: https://proceedings.mlr.press/v202/kwon23b.html.

21 Jerry Li and Ludwig Schmidt. Robust and proper learning for mixtures of Gaussians via systems of polynomial inequalities. In *COLT*, Proceedings of Machine Learning Research, pages 1302–1382. PMLR, 2017. URL: http://proceedings.mlr.press/v65/li17a.html.

22 Amit Sahai and Salil Vadhan. A complete problem for statistical zero knowledge. *J. ACM*, 50(2):196–249, 2003. doi:10.1145/636865.636868.

Testing k-Submodularity

Themistoklis Haris
Department of Computer Science, Boston University, MA, USA

Diptaksho Palit
Department of Computer Science, Boston University, MA, USA

Abstract

We initiate the study of property testing for k–submodular functions, a higher-dimensional analogue of submodular functions defined on partial partitions of a ground set. While k–submodularity retains the diminishing-returns flavor of ordinary submodularity, it also introduces a pairwise monotonicity constraint comparing competing assignments of the same element. This additional local structure makes the testing problem qualitatively different from the classical case.

Our results show a sharp contrast between distance regimes. In the ℓ_p regime for $p \geq 1$, we prove that every bounded k–submodular function is close to a junta on the hypergrid. Combined with an implicit-learning tester for hypergrid domains, this yields a constant-query tester for k–submodularity. In the Hamming distance regime, k–submodularity admits two qualitatively different local witnesses – violated *squares* for diminishing marginal gains, and violated *triangles* for pairwise-monotonicity failures – and the latter has no counterpart at $k = 1$. We prove density theorems for both witness types via repair on filters and ideals of partial partitions, yielding non-adaptive, one-sided sub-exponential-query testers for the two component properties of k–submodularity. We then exhibit a configuration in which the two repair directions are forced into opposition on a shared vertex, identifying a structural barrier to combining these into a tester for the full property.

Finally, for bounded-range functions, we give an adaptive tester for monotone k–submodularity via a pseudo-DNF representation and learning on the hypergrid. Several of the structural and learning tools developed here may be useful for testing other properties over product domains.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases property testing, sublinear algorithms, submodular functions

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.52

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/pdf/2606.30433` [9]

Funding *Diptaksho Palit*: Supported by the U.S. National Science Foundation under Grant No. 2022446.

Acknowledgements T.H. worked on k-submodularity testing as a final project for the 2024 Sublinear Algorithms course taught by Ronitt Rubinfeld. T.H. and D.P. started working on this project for a course on Sublinear Algorithms taught by Sofya Raskhodnikova [14]. We thank Fabian Spaeh for inspiring us to investigate this question through his work on k-submodular optimization.

1 Introduction

Submodularity is one of the canonical examples of a global property with a simple local interpretation. A set function is submodular when it satisfies the principle of *diminishing marginal returns*: the incremental value of adding an element can only decrease as the set to which it is added grows. Formally, a set function $f : 2^{[n]} \to \mathbb{R}$ is said to be *submodular* if, for all subsets $X, Y \subseteq [n]$, it satisfies the inequality:

$$f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y).$$

© Themistoklis Haris and Diptaksho Palit;
licensed under Creative Commons License CC-BY 4.0

This local-to-global nature makes submodularity a natural target for property testing [8], where one is given oracle access to an unknown function and must distinguish functions satisfying a property from those that are far from satisfying it. Yet even for ordinary submodularity, the query complexity of testing remains poorly understood. Seshadhri and Vondrák [16] gave a tester using roughly $(1/\varepsilon)^{\sqrt{n}}$ queries and proved a linear lower bound; the latter was later improved to a nearly quadratic lower bound by Hatami and Vondrák [10]. The resulting exponential gap has remained open, and has motivated the study of more structured regimes, such as bounded-range submodular functions [15] and testing under ℓ_p distance rather than Hamming distance [1].

In this work we ask what happens to this picture for a higher-dimensional analogue of submodularity. For a fixed integer $k \geq 1$, a point in $[k+1]^n$ can be viewed as a *partial partition* of the ground set $[n]$: each element is either assigned to one of k parts, or left unassigned. A function on these partial partitions is k–*submodular* if marginal gains diminish no matter which part an element is assigned to. The case $k = 1$ is ordinary submodularity, while already for $k = 2$ the domain has multiple incompatible directions in which a single element may be added. This function class has been studied extensively in optimization [12, 4], with applications including tensor placement [12] and online ad allocation [5].

From a testing perspective, k–submodularity is not merely submodularity on a larger domain. A useful characterization, due to Ward and Živný [17], decomposes k–submodularity into two conditions. The first is *diminishing marginal gains*, which generalizes the familiar submodular inequality inside each direction of the hypergrid. The second is *pairwise monotonicity*, a constraint comparing the marginal gains of assigning the same element to two different parts. One way to understand this decomposition is to separate two kinds of local interactions. When two partial partitions make compatible choices, k–submodularity behaves like ordinary diminishing returns along the corresponding direction. When they make incompatible choices for the same element, the join operation discards that element from both parts; the resulting inequality says that assigning the element to two different parts cannot have two strongly negative marginal gains at the same time. This second condition has no analogue when $k = 1$, and it is the source of much of the new combinatorial structure in the testing problem.

We initiate the study of testing k–submodularity. Our results show a sharp contrast between two distance regimes. In the ℓ_p regime, analytic structure survives the passage from the Boolean hypercube to the hypergrid: we prove a junta-approximation theorem for k–submodular functions and combine it with an implicit-learning tester on hypergrids. In the Hamming distance regime, the local structure fragments: we obtain testers for the two local constraints by developing square and triangle witness structures, and show that their repair procedures expose a genuine obstruction to the standard local-repair paradigm for full k–submodularity. Along the way, we develop testing and learning tools for hypergrid domains that may be useful beyond this problem.

1.1 Our Results

We organize our contributions around three questions. First, does the analytic structure that makes submodularity testable in ℓ_p distance survive on the richer domain of partial partitions? Second, in the Hamming distance regime, can the local-violation framework for submodularity be adapted to the two distinct constraints that characterize k–submodularity? Third, can bounded range assumptions recover efficient testability for a meaningful subclass of k–submodular functions?

ℓ_p testing from hypergrid junta structure

Our first result is a constant-query tester for k–submodularity in ℓ_p distance. Here and throughout this section, constant means independent of the ambient dimension n.

▶ **Theorem 1.** *For all $p \geq 1$, there exists an adaptive, one-sided error ℓ_p-tester for k-submodularity making $q = \left(\frac{p}{\varepsilon}\right)^2 \cdot 2^{\left(\frac{p}{\varepsilon}\right)^2 \log\left(\frac{p}{\varepsilon}\right)}$ queries.*

The main structural input is not a black-box reduction to the submodular case. We prove that every bounded k–submodular function is close to a function depending on only a small number of coordinates, even though each coordinate can move in k competing directions. This requires choosing influential coordinates in a way that respects all parts of a partial partition, and uses the fact that k–submodular functions satisfy a self-bounding variance inequality.

▶ **Theorem 2.** *Let $f : [k+1]^J \to [0,1]$ be a k–submodular function and $\varepsilon > 0$ be a small enough constant. Then there exists a k–submodular function $h : [k+1]^J \to [0,1]$ that depends only on $J' \subseteq J$ dimensions, where*

$$|J'| = O\left(\frac{1}{\varepsilon^2} \cdot \log \frac{1}{\varepsilon}\right),$$

and the function h satisfies

$$\|f - h\|_2 \leq \varepsilon.$$

We then develop the testing component needed to turn this structure into an algorithm. The implicit-learning framework of Blais and Bommireddi [1] relies on identifying influential coordinates and checking consistency with a small junta. We formulate and analyze this framework on hypergrid domains using Fourier analysis over generalized product spaces, and simplify parts of the original argument by separating the learning and final testing samples.

▶ **Theorem 3.** *Let $p \geq 1$. Suppose $\mathcal{P} \subseteq \{f : [k+1]^n \to [0,1]\}$ is a property of functions such that for any $f \in \mathcal{P}$ there exists a κ-junta $h : [k+1]^n \to [0,1]$ satisfying $||f - h||_2 \leq \varepsilon$. Then, there exists a tester that, given query access to an unknown function f, will make $\varepsilon^{-9} \cdot 2^{O(\kappa \log \kappa)}$ queries to f and with probability at least $2/3$ accept if $f \in \mathcal{P}$ and reject if f is ε-far from $\mathcal{P}$ in the ℓ_2 norm.*

Local testing in Hamming distance

Our second set of results addresses the Hamming-distance regime, where the Seshadhri–Vondrák local-repair paradigm for ordinary submodularity does not extend directly. Pairwise monotonicity is vacuous when $k = 1$, but on the hypergrid its violations form a new witness shape – *triangles* – incomparable with the *squares* that witness diminishing-marginal-gain failures. Repair likewise shifts from Boolean subcubes to filters and ideals of partial partitions, and the density arguments must be re-proved in this setting. We obtain non-adaptive, one-sided testers for the two component properties of k–submodularity:

▶ **Theorem 4.** *Let $f : [k+1]^n \to \mathbb{R}$ be given via oracle access. There exist non-adaptive, one-sided error testers for the diminishing marginal gains and pairwise monotonicity properties with the following guarantees:*

- *If f satisfies the property, the tester accepts with probability 1.*
- *If f is ε-far from the property (i.e., at least $\varepsilon \cdot [k+1]^n$ values must be modified to satisfy it), the tester rejects with probability at least $2/3$.*

The tester for diminishing marginal gains makes $q = 4n^2 \cdot (1/\varepsilon)^{\Theta(\sqrt{n \log n})}$ *queries; the tester for pairwise monotonicity makes* $q = 3kn \cdot (1/\varepsilon)^{\Theta(\sqrt{n \log n})}$ *queries.*

Notably, only the pairwise-monotonicity tester has query complexity depending on k, despite k governing the structure of the domain.

Combined, these results expose a structural difference between k–submodularity and ordinary submodularity that the analytic (ℓ_p) regime does not see. In Section 4.3 we exhibit an explicit configuration (Figure 1) on a high item-weight filter in which repairing a triangle violation forces value changes in the *opposite* direction from those needed to repair an overlapping square violation, on a shared vertex. The two repair directions are not merely incompatible at one point – they are jointly inconsistent across an entire region. We record this as Observation 32: it rules out the natural component-wise extension of local repair to full k–submodularity, and isolates a concrete barrier that any sub-exponential local-witness tester must overcome by enlarging the witness vocabulary beyond squares and triangles.

Bounded range and monotone k–submodularity

Finally, we show that efficient Hamming-distance testing can be recovered for monotone k–submodular functions with bounded range. The result is again obtained through learning, but the relevant representation is no longer a junta. Instead, we utilize pseudo-DNF representations for functions on the hypergrid and prove that bounded-range monotone k–submodular functions have bounded-width pseudo-DNFs.

▶ **Theorem 5.** *Let* $\varepsilon, \delta \in (0,1)$ *be constants. There exists a tester* $\mathcal{T}$ *which, given query access to a function* $f : [k+1]^n \to \{0, 1, \cdots, r\}$ *satisfies the following.*

1. *If* f *is monotone* k*–submodular, then* $\mathcal{T}$ **accepts** *with probability* $\geq 1 - \delta$.
2. *If* f *is* ε*-far from monotone* k*–submodularity, then* $\mathcal{T}$ **rejects** *with probability* $\geq 1 - \delta$.
3. $\mathcal{T}$ *makes at most* $\mathsf{poly}(n, 1/\varepsilon, 1/\delta, (kr)^{O(rk^2 \log(r/\varepsilon))})$ *queries.*

The learning result for bounded-width pseudo-DNFs uses a hypergrid version of the switching-lemma/Fourier-sparsity route and yields a tester via the Kushilevitz-Mansour algorithm. We expect this learner to be useful beyond k–submodularity.

1.2 Prior Work

Testing whether a given function is submodular was first examined by Parnas, Ron and Rubinfeld [13], who studied the limited setting of testing for a class of matrices called *Monge matrices*. In its full generality, the problem was considered by Seshadhri and Vondrák [16]. They developed the local-witness and repair framework for Hamming-distance testing on the Boolean hypercube, proving an upper bound of $(1/\varepsilon)^{O(\sqrt{n \log n})}$ and a linear lower bound. The latter was later improved to $\Omega(n^2/\log n)$ by Hatami and Vondrák [10]. Blais and Bommireddi [1] subsequently studied the problem in the ℓ_p norm, and using an approximation result from [6], designed an implicit-learning tester with constant query complexity in the input length. Raskhodnikova and Yaroslavtsev [15] studied bounded-range submodular functions through a learning-based approach and proved a polynomial-time upper bound in this regime. Chakrabarty and Huang [3] studied the problem of testing for coverage functions, which are a special type of submodular function.

Our work is informed most directly by the three submodularity-testing frameworks of Seshadhri and Vondrák [16], Blais and Bommireddi [1], and Raskhodnikova and Yaroslavtsev [15]. These works provide the conceptual starting points for the three regimes we study.

The passage to k–submodularity introduces new difficulties in each case: local witnesses split into squares and triangles in the Hamming regime, influential coordinates must account for competing assignments in the ℓ_p regime, and bounded-range learning requires representations over the hypergrid rather than the Boolean cube.

To our knowledge, there is no prior work on testing k-submodularity, though the property has been studied extensively from the optimization perspective [12, 4]. We also use the structural characterization of Ward and Živný [17], which identifies k–submodularity with the conjunction of diminishing marginal gains and pairwise monotonicity.

1.3 Technical Overview

The main technical theme is that the hypergrid introduces several inequivalent notions of locality. In the ℓ_p regime, locality is analytic and is captured by influential coordinates. In the Hamming regime, locality is combinatorial and is captured by small violated configurations. For bounded-range functions, locality appears through compact symbolic representations. In each part of the paper we develop the corresponding notion of locality for partial partitions.

The analytic regime: finding the right influential coordinates

The ℓ_p tester has two ingredients: a junta-approximation theorem for k–submodular functions, and a tester for properties that are close to juntas on the hypergrid. The obstacle in the first ingredient is that a coordinate has k possible nonzero directions. Several natural ways of declaring a coordinate influential in one part at a time almost work, but fail to interact correctly with the boosting lemma for down-monotone families (Lemma 12).

Our solution is to define the relevant influence threshold by looking across all partitions of the remaining coordinates. This lets the selection rule remain compatible with the combinatorial boosting argument while still respecting the multi-directional nature of the domain. The second key input is a variance bound: we prove that k–submodular functions are 2-self bounding (Lemma 14). Together, these ingredients yield a junta depending on only $O(\varepsilon^{-2}\log(1/\varepsilon))$ coordinates.

Once the junta structure is established, the testing step is conceptually separate. We analyze implicit learning directly on $[k+1]^n$ using Fourier analysis over product domains. The tester identifies a candidate set of influential coordinates and then checks whether the observed behavior is consistent with a junta from the target property. Using an independent sample in the final testing stage also simplifies the decorrelation issues that arise in the original Boolean-domain analysis of [1].

The Hamming regime: local witnesses and repair

In Hamming distance, a tester needs many local witnesses whenever the function is far from the property. For ordinary submodularity, the relevant witness is a violated square in the Boolean hypercube, and the analysis of Seshadhri and Vondrák [16] shows that few violated squares imply a global repair.

For k–submodularity, a single witness shape is no longer sufficient. Using the characterization of Ward and Živný [17], we separate the property into diminishing marginal gains and pairwise monotonicity. Diminishing marginal gains is witnessed by generalized violated squares, while pairwise monotonicity is witnessed by violated triangles. We prove density statements for both witnesses by designing repair procedures on large inclusion-closed regions of the hypergrid: ideals $\{Y \preceq X\}$ and filters $\{Y \succeq X\}$. This gives sub-exponential testers for the two component properties.

The same repair viewpoint also yields a structural observation about the full Hamming-distance problem (Observation 32, illustrated in Figure 1). On high item-weight filters, the repair direction needed for a violated square is forced into conflict with the repair direction needed for an overlapping violated triangle on a shared vertex. This is not a limitation of our proof – it is a structural property of the problem: any tester built on the squares-and-triangles vocabulary, no matter how the repair is organized, must encounter a configuration of this kind. A sub-exponential tester for full k–submodularity in Hamming distance therefore requires an enriched witness vocabulary, not merely a more clever combination of square and triangle repairs.

The bounded-range regime: representation before testing

For bounded-range monotone k–submodular functions, we take a representation-theoretic route. We first show that such functions admit bounded-width pseudo-DNF representations over the hypergrid. This representation captures the monotone threshold-like structure imposed jointly by bounded range and diminishing marginal gains.

We then prove a hypergrid switching lemma for pseudo-DNFs, using low-depth k-ary decision trees in place of binary trees. Combined with Fourier analysis on the hypergrid, this implies sparse Fourier approximation for bounded-width pseudo-DNFs. The resulting learner, together with the Kushilevitz-Mansour algorithm [11], yields the bounded-range tester.

1.4 Discussion and Open Problems

Our results leave several natural questions open. The most immediate is whether full k–submodularity can be tested in Hamming distance with sub-exponential query complexity. Our component testers for diminishing marginal gains and pairwise monotonicity show that each constraint has many local witnesses when violated, but the repair procedures for the two constraints can be forced to conflict. Resolving this – either by exhibiting a richer witness family with a compatible repair procedure, or by proving a sub-exponential lower bound that confirms the barrier is fundamental – is, in our view, the right next step in this regime.

A second direction is to understand the bounded-range setting without monotonicity. Our tester in this regime relies on a pseudo-DNF representation for monotone k–submodular functions, and it is unclear whether a comparable representation exists for non-monotone functions.

Finally, it would be interesting to prove lower bounds for testing k–submodularity. The known lower bounds for submodularity do not transfer directly, since natural embeddings can destroy pairwise monotonicity. Thus even formulating the right hard instances appears to require using the genuinely multi-part structure of the domain.

1.5 Organization

The paper is organized as follows. In Section 2, we formally define the problem and related structures on the hypergrid. In Section 3, we prove the junta approximability theorem for k–submodular functions. In Section 4, we develop Hamming-distance testers for *diminishing marginal gains* and *pairwise monotonicity*, and analyze the obstruction to combining their repairs into a local-repair tester for full k–submodularity. We refer to the full version of the paper [9] for the analysis of the hypergrid implicit-learning theorem and construction of our ℓ_0 tester for monotone k–submodularity in the bounded range regime.

2 Preliminaries

▶ **Definition 6** (Partial partition). *A* partial partition *of* $[n]$ *into* k *subsets is defined* $X = (X_1, ..., X_k)$*, where each* $X_i \subseteq [n]$ *and* $X_i \cap X_j = \emptyset$ *for all* $i \neq j \in [k]$*. We let* $X_{k+1} = [n] \setminus \cup_{i=1}^{k} X_i$*, and this allows us to associate* partial partitions *of* $[n]$ *into* k *subsets with the hypergrid* $[k+1]^n$ *(each element of* $[n]$ *is associated to a bucket in* $[k+1]$*, yielding a vector in* $[k+1]^n$*). If* $e \notin X_{k+1}$*, we say that* $e \in X$*. We impose a* partial order *on* partial partitions *by defining* $X \preceq Y$ *if and only if* $X_i \subseteq Y_i$ *for all* $i \in [k]$*. We let* $|X|$ *denote the* item-weight *of a partial partition, which is the total number of items picked. Numerically, this is equal to* $\sum_{i=1}^{k} |X_i|$*.*

▶ **Definition 7** (Marginal gains). *Define the* marginal gains *of a function* f *on a partial partition* X *with respect to an index* $i \in [k]$ *and an element* $e \in X_{k+1}$ *as*

$$\Delta_f(X, i, e) := f(X_1, ..., X_i \cup \{e\}, ..., X_k) - f(X).$$

Letting $X_e^i := (X_1, ..., X_i \cup \{e\}, ..., X_k)$*, we can rewrite this as* $\Delta_f(X, i, e) = f(X_e^i) - f(X)$*.*

▶ **Definition 8** (k–submodularity). *A function* $f : [k+1]^n \to \mathbb{R}$ *is* k–submodular *if, for all partial partitions* X, Y*, the following holds.*

$$f(X) + f(Y) \geq f(X \sqcup Y) + f(X \sqcap Y),$$

where $X \sqcap Y := (X_1 \cap Y_1, \cdots, X_k \cap Y_k)$ *and*

$$X \sqcup Y = \left((X_1 \cup Y_1) \setminus \bigcup_{i \neq 1} (X_i \cup Y_i), \cdots, (X_k \cup Y_k) \setminus \bigcup_{i \neq k} (X_i \cup Y_i) \right).$$

The following theorem of Ward and Živnỳ [17] gives us an equivalent definition of k–submodularity, one that is described by two more intuitive properties. The intuition is that every local comparison between two partial partitions can be decomposed into compatible and conflicting moves. Compatible moves are governed by the usual diminishing-returns principle: adding an unassigned element to part i becomes less valuable as the current partial partition grows. Conflicting moves arise only when the same element could be assigned to two different parts. For example, comparing X_e^i and X_e^j gives $X_e^i \sqcap X_e^j = X$ and $X_e^i \sqcup X_e^j = X$, so the k–submodular inequality becomes

$$\Delta_f(X, i, e) + \Delta_f(X, j, e) \geq 0.$$

Thus pairwise monotonicity is the local condition that prevents two competing assignments of the same element from both being too costly.

▶ **Theorem 9** ([17]). *A function* $f : [k+1]^n \to \mathbb{R}$ *is* k–submodular *if and only if the following two conditions hold.*

- ***Diminishing marginal gains.*** *For all partial partitions* X, Y *such that* $X \preceq Y$*,* $e \notin Y$*, and* $i \in [k]$*, the following holds.*

$$\Delta_f(X, i, e) \geq \Delta_f(Y, i, e).$$

- ***Pairwise monotonicity.*** *For all partial partitions* X*,* $e \notin X$*, and* $i \neq j \in [k]$*, the following holds.*

$$\Delta_f(X, i, e) + \Delta_f(X, j, e) \geq 0.$$

3 Junta approximation of k–submodular functions

In the following two sections, we prove the structural and algorithmic ingredients behind our ℓ_p tester for k–submodularity. The structural ingredient is a junta approximation theorem for k–submodular functions on the hypergrid. The main difficulty is choosing influential coordinates in a way that accounts for all k directions of a coordinate while still giving a dimension-independent approximation.

This approximation theorem, combined with an implicit learning tester for properties on hypergrids with a good junta approximation, yields the desired tester.

We prove the following approximation theorem.

▶ **Theorem 10** (Reminder of Theorem 2). *Let $f : [k+1]^J \to [0,1]$ be a k–submodular function and $\varepsilon > 0$ be a small enough constant. Then there exists a k–submodular function $h : [k+1]^J \to [0,1]$ that depends only on $J' \subseteq J$ dimensions, where*

$$|J'| = O\left(\frac{1}{\varepsilon^2} \cdot \log \frac{1}{\varepsilon}\right),$$

and the functions h satisfies

$$\|f - h\|_2 \leq \varepsilon.$$

Before we get to the proof of Theorem 2, we first set up some necessary machinery.

3.1 Setup

There are three main parts to the setup.

Boosting lemma. We cite the following boosting lemma due Goemans and Vondrák [7].

▶ **Definition 11** (Down-monotone family). *Let X be a finite set and $\mathcal{F} \subseteq 2^X$ be a collection of subsets of X. The family $\mathcal{F}$ is called down-monotone if the following condition holds.*

$$A \in \mathcal{F} \wedge B \subseteq A \implies B \in \mathcal{F}.$$

▶ **Lemma 12** ([7] Boosting lemma). *Let X be a finite set and $\mathcal{F} \subseteq 2^X$ be a down-monotone family. For $p \in (0,1)$, define*

$$\sigma_p := \Pr[X(p) \in \mathcal{F}],$$

where $X(p)$ is a random subset of X such that each element is picked independently with probability p. Then

$$\sigma_p = (1-p)^{\phi(p)},$$

where $\phi(p)$ is a non-decreasing function for $p \in (0,1)$.

Self–bounding functions and concentration. A very useful tool in bounding the variance of submodular functions comes from the literature of concentration for *self–bounding* functions.

▶ **Definition 13** (Self–bounding function). *A function $f : \mathcal{X}^n \to [0,1]$ is **α-self–bounding** if there exist functions $f_i : \mathcal{X}^{n-1} \to [0,1]$ such that for all $x_1, \cdots, x_n \in \mathcal{X}$ and all $i \in [n]$, the following conditions hold.*

1. $0 \leq f(x_1, \cdots, x_n) - f_i(x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_n) \leq 1$; *and*
2. $\sum_{i=1}^{n} (f(x_1, ..., x_n) - f_i(x_1, x_2, ..., x_{i-1}, x_{i+1}, ..., x_n)) \leq \alpha f(x_1, ..., x_n)$.

▶ **Lemma 14** ([2] Bounded variance lemma). *If a function $f : \mathcal{X}^n \to [0,1]$ is α-self–bounding then*

$$\mathrm{Var}[f(X)] \leq \alpha \cdot \mathbb{E}[f(X)].$$

Bounded variance of k–submodular functions. Submodular functions that are Lipschitz are self–bounding, which allows us to bound their variance. We show that a similar property holds for k–submodular functions.

▶ **Definition 15** (Lipschitz function). *A function $f : [k+1]^n \to [0,1]$ is α-Lipschitz if $|\Delta_f(X,j,i)| \leq \alpha$ for all $i \in [n]$ and $j \in [k]$.*

▶ **Lemma 16.** *Let $f : [k+1]^n \to [0,1]$ be 1–Lipschitz and k–submodular. Then:*

$$\mathrm{Var}[f] \leq 2 \cdot \mathbb{E}[f].$$

Proof. We prove this by showing that f is 2-self–bounding, and invoking Lemma 14. For $i \in [n]$, define the function f_i as

$$f_i(x_1, \cdots, x_{i-1}, x_{i+1}, \cdots x_n) := \min_{j \in [k+1]} \{f(x_1, \cdots, x_{i-1}, j, x_{i+1}, x_n)\}.$$

Then the first condition

$$0 \leq f(x_1, ..., x_n) - f_i(x_1, ..., x_{i-1}, x_{i+1}, ..., x_n) \leq 1$$

holds trivially (by choice of minimum value and the range of f being $[0,1]$). By pairwise monotonicity of f, the following holds.

$$\begin{aligned} &f(x_1, ..., x_n) - f_i(x_1, ..., x_{i-1}, x_{i+1}, ..., x_n) \\ &\qquad \leq 2 \cdot \left(f(x_1, ..., x_n) - f(x_1, ..., x_{i-1}, k+1, x_{i+1}, ..., x_n)\right). \end{aligned}$$

Here, $x_i = k+1$ corresponds to item i not being put into any bucket. Finally, the following holds for all $i \in [n]$ due to the marginal gains property.

$$\begin{aligned} &f(x_1, ..., x_n) - f(x_1, ..., x_{i-1}, k+1, x_{i+1}, ..., x_n) \\ &\leq f(x_1, \cdots, x_i, k+1, \cdots, k+1) - f(x_1, \cdots, x_{i-1}, k+1, \cdots, k+1). \end{aligned}$$

This yields the desired condition.

$$\begin{aligned} &\sum_{i=1}^{n} (f(x_1, \cdots, x_n) - f_i(x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_n)) \\ &\qquad \leq \sum_{i=1}^{n} 2 \cdot ((f(x_1, \cdots, x_n) - f(x_1, \cdots, x_{i-1}, k+1, x_{i+1}, \cdots, x_n)) \\ &\qquad \leq \sum_{i=1}^{n} 2 \cdot (f(x_1, \cdots, x_i, k+1, \cdots) - f(x_1, \cdots, x_{i-1}, k+1, \cdots)) \\ &\qquad \leq 2 \cdot f(x_1, \cdots, x_n). \end{aligned}$$

We have thus shown that f is 2-self–bounding, and by Lemma 16, $\mathrm{Var}[f] \leq 2 \cdot \mathbb{E}[f]$. ◀

3.2 Weak junta approximation

We first prove the following "weak" junta approximation for k–submodular functions.

▶ **Theorem 17** (Weak junta approximation). *Let $f : [k+1]^J \to [0,1]$ be a k–submodular function and $\varepsilon \in (0,1)$ be a constant. Then there exists a k–submodular function $h : [k+1]^J \to [0,1]$ that depends only on $J' \subseteq J$ dimensions, where*

$$|J'| \leq \frac{32}{\varepsilon^2} \cdot \log \frac{8|J|}{\varepsilon^2},$$

and the function h satisfies

$$\|f - h\|_2 := \left(\mathbb{E}_{x \in [k+1]^J} \left[(f(x) - h(x))^2\right]\right)^{\frac{1}{2}} \leq \frac{1}{2} \cdot \varepsilon.$$

We prove Theorem 17 by first describing a procedure to construct the set J' and then showing that there exists a J'-junta satisfying the desired properties.

Let $S \subseteq J$ and $P : J \to [k+1]^n$ be some partition function over J. We denote by S_P the point on the hypergrid obtained by placing each $i \in S$ into bucket $P(i)$. We denote by $S(\delta)_P$ a random set generated by sampling each element in S independently with probability δ.

Algorithm 1 Identifying the junta set J'.

Input: Oracle access to k–submodular function $f : [k+1]^J \to [0,1]$; parameters $\alpha, \delta > 0$

$S \leftarrow \emptyset$.

while there exists $i \notin S$ such that for all partitions P of J we have $\Pr\left[\Delta_f(S(\delta)_P, P(i), i) > \alpha\right] > \frac{1}{2}$ **do**

    $S \leftarrow S \cup \{i\}$

return $J' = S$.

First we show that the number of indices picked by Algorithm 1 is small.

▷ Claim 18. The set J' output by Algorithm 1 satisfies $|J'| \leq \frac{2}{\alpha\delta}$.

Proof. Let P be an arbitrary partition of J, and X be an arbitrary subset of J. For any $i \in X$, let $X_{<i}$ denote the set of items in X that were added to S before i in Algorithm 1. Define

$$X_P^+ := \{i \in X \mid \Delta_f(X_{<i,P}, P(i), i) > \alpha\}.$$

Then by *diminishing marginal gains*, at the end of the execution of Algorithm 1, we have

$$\begin{aligned}\mathbb{E}[f(S(\delta)_P^+)] &= f(\emptyset) + \sum_{i \in S(\delta)_P^+} \Delta_f(S(\delta)_{<i,P}^+, P(i), i) \\ &> \mathbb{E}[|S(\delta)_P^+|] \cdot \alpha. && \text{[Assuming } f(\emptyset) = 0 \text{ WLOG]}\end{aligned}$$

Next, we show that $\mathbb{E}[|S(\delta)_P^+|] \geq \frac{\delta \cdot |S|}{2}$. If that is true, then

$$1 \geq \mathbb{E}[f(S(\delta)_P^+)] > \frac{\delta \cdot |S| \cdot \alpha}{2} \implies |S| \leq \frac{2}{\delta\alpha}.$$

To prove our claim, we observe that by construction, for all $u \in S$,

$$\Pr[u \in S(\delta)_P^+] = \Pr[u \in S(\delta)] \cdot \Pr[\Delta_f(S(\delta)_{<u,P}, P(u), u) > \alpha] > \frac{\delta}{2}.$$

Writing $|S(\delta)_P^+| = \sum_{u \in S} X_u$, where X_u is the indicator for whether $u \in S(\delta)_P^+$, we get

$$\mathbb{E}[|S(\delta)_P^+|] = \sum_{u \in S} \Pr[X_u] \geq \frac{\delta \cdot |S|}{2}.$$ ◁

Next, we show that for all partitions P and any "unimportant" coordinate $i \in J \setminus J'$, the probability of the marginal gains on top of J' being large with respect to P when adding i to $P(i)$ is very small if we sample elements from J' with probability $\frac{k}{k+1}$ rather than δ.

▷ Claim 19. For all $i \in J \setminus J'$, all $j \in [k]$, and all partitions $P : J' \to [k]$, the following holds.

$$\Pr_{J'(\frac{k}{k+1})}\left[\Delta_f\left(J'\left(\frac{k}{k+1}\right)_P, j, i\right) > \alpha\right] \leq \left(\frac{1}{k+1}\right)^{\frac{1}{2\delta}}.$$

Proof. Consider the following family of points

$$\mathcal{F} = \{X \in [k+1]^n \mid X \preceq J'_P \text{ and } \Delta_f(X, j, i) > \alpha\}$$

This is a down-monotone set – if $Y \preceq X \in \mathcal{F}$ then $Y \in \mathcal{F}$. This is due to *diminishing marginal gains* – $\Delta_f(Y, j, i) \geq \Delta(X, j, i)$. If we let $\sigma_p := \Pr[J'(p)_P \in \mathcal{F}]$ for $p \in (0, 1)$, then by Lemma 12,

$$\sigma_p = (1-p)^{\phi(p)},$$

for some non-decreasing function ϕ. The termination condition of Algorithm 1 guarantees that

$$\sigma_\delta := \Pr[J'(\delta)_P \in \mathcal{F}] = (1-\delta)^{\phi(\delta)} \leq \frac{1}{2},$$

which implies $\phi(\delta) \geq \frac{1}{2\delta}$ for $\delta \leq \frac{1}{2}$. Since ϕ is non-decreasing, we have

$$\phi\left(\frac{k}{k+1}\right) \geq \phi(\delta) \geq \frac{1}{2\delta} \implies$$
$$\sigma_{\frac{k}{k+1}} = \left(1 - \frac{k}{k+1}\right)^{\phi\left(\frac{k}{k+1}\right)} \leq \left(\frac{1}{k+1}\right)^{1/2\delta}. \qquad \triangleleft$$

We now define our junta function. Let $\bar{J} := J \setminus J'$. Define $h : [k+1]^n \to [0, 1]$ as

$$h(X) := \mathop{\mathbb{E}}_{Y \in [k+1]^{\bar{J}}}[f(X_{J'}, Y)], \tag{1}$$

where $x_{J'}$ is the point in the hypergrid where the coordinates in J' are assigned as they were in J and the coordinates in $\bar{J}$ are non-assigned.

First, we show that h is k–submodular.

▷ Claim 20. The function h defined in Equation (1) is k–submodular.

Proof. We have to show that h satisfies the two following conditions.

1. *Diminishing marginal gains.* Let $X, Z \in [k+1]^n$ with $Z \preceq X$ and $e \notin X$. Let $j \in [k]$. Then, we have

$$\Delta_h(Z, j, e) = h(Z^i_e) - h(Z) = \begin{cases} 0, & \text{if } j \notin J' \\ \mathop{\mathbb{E}}_{Y \in [k+1]^{\bar{J}}}[f((Z^i_e)_{J'}, Y) - f(Z_{J'}, Y)], & \text{otherwise} \end{cases}.$$

If $j \notin J'$, then $\Delta_h(X, j, e) = 0$ as well, so the diminishing marginal gains condition holds. Otherwise, by the k–submodularity of f, we know that for all $y \in [k+1]^{\bar{J}}$,

$$f((Z^i_e)_{J'}, Y) - f(Z_{J'}, Y) \geq f((X^i_e)_{J'}, Y) - f(X_{J'}, Y)$$

which allows us to conclude that $\Delta_h(Z, j, e) \geq \Delta_h(X, j, e)$, and thus that h satisfies diminishing marginal gains.

2. *Pairwise Monotonicity.* Let $X \in [k+1]^n$, $i \neq j \in [k]$ and $e \in X_{k+1}$. If $e \notin J$, then $\frac{1}{2}\left(h(X^i_e) + h(X^j_e)\right) = h(X)$. Otherwise, by the pairwise monotonicity of f,

$$\frac{1}{2}\left(h(X^i_e) + h(X^j_e)\right) = \mathop{\mathbb{E}}_{Y\in[k+1]^{\bar{J}}} \frac{1}{2}[f((X^i_e)_{J'}, Y) - f((X^j_e)_{J'}, Y)] \geq \mathop{\mathbb{E}}_{Y\in[k+1]^{\bar{J}}} [f(X_{J'}, Y)] = h(X).$$

We can thus conclude by Theorem 9 that h is k–submodular. ◁

Our final step is to show that h is a good approximation to f in the ℓ_2-norm.

▶ **Definition 21** (Bad points). *A point $X_{J'}$ in the restricted hypergrid is called* bad *if there exists $i \in \bar{J}$ and $j \in [k]$ such that*

$$\Delta_f(X_{J'}, j, i) > \alpha,$$

or there exists $i \in X_{J'}$

$$\Delta_f(X_{J'} \setminus i, X_{J'}(i), i) < -\alpha.$$

The ℓ_2 distance can be bounded as below.

$$\begin{aligned}
\|h - f\|_2^2 &= \mathop{\mathbb{E}}_{X\in[k+1]^J}[(f(X) - h(X))^2] \\
&= \mathop{\mathbb{E}}_{X_{J'}\in[k+1]^{J'}}\left[\mathop{\mathbb{E}}_{Y\in[k+1]^{\bar{J}}}\left[(f(X_{J'}, Y) - \mathop{\mathbb{E}}_{Y'\in[k+1]^{\bar{J}}}[f(X_{J'}, Y')])^2\right]\right] \\
&= \mathop{\mathbb{E}}_{X_{J'}\in[k+1]^{J'}}\left[\mathop{\mathrm{Var}}_{Y\in[k+1]^{\bar{J}}}[f(X_{J'}, Y)]\right] \\
&\leq \Pr[X_{J'} \text{ is } \textit{bad}] + \mathop{\mathbb{E}}_{X_{J'}\in[k+1]^{J'} \text{ good}}\left[\mathop{\mathrm{Var}}_{Y\in[k+1]^{\bar{J}}}[f(X_{J'}, Y)]\right]. \qquad \text{[Because } f(X) \leq 1\text{]}
\end{aligned}$$

We first show that for appropriate parameters to Algorithm 1, the probability that a point is bad is small.

▷ Claim 22. If we set $\delta = \dfrac{\log(k+1)}{2\log(8 \cdot k \cdot |J| \cdot \varepsilon^{-2})}$ in Algorithm 1, then

$$\Pr_{X_{J'}\in[k+1]^{J'}}[X_{J'} \text{ is } \textit{bad}] \leq \frac{\varepsilon^2}{8}.$$

Proof. Recall that we consider a uniform distribution over $X_{J'} \in [k+1]^{J'}$. This distribution is the same as sampling a partition $P : J' \to [k]$ uniformly at random and returning $J'\left(\frac{k}{k+1}\right)_P$. To see this, fix an arbitrary point $X \in [k+1]^{J'}$. Let $A := \{i \in J' : x_i = k+1\}$ and $B = J' \setminus A$. The probability of generating this particular X with the latter sampling procedure is

$$\Pr[X] = \left(\frac{1}{k+1}\right)^{|A|} \cdot \left(\frac{k}{k+1}\right)^{|B|} \cdot \left(\frac{1}{k}\right)^{|B|} = \frac{1}{(k+1)^{|J'|}}$$

So, via a union bound, Claim 19 and by plugging in $\delta = \frac{\log(k+1)}{2\log(8\cdot k\cdot |J|\cdot \varepsilon^{-2})}$, we get that

$$\begin{aligned}\Pr[X_{J'} \text{ is } bad] &= \Pr_{J'\left(\frac{k}{k+1}\right),P:J'\to[k]}\left[\bigcup_{j\in[k]}\bigcup_{i\in\bar{J}}\left(\Delta_f\left(J'\left(\frac{k}{k+1}\right)_P, j, i\right) > \alpha\right)\right] \\ &\leq |\bar{J}|\cdot k\cdot\left(\frac{1}{k+1}\right)^{1/2\delta} \\ &\leq \frac{\varepsilon^2}{8}.\end{aligned}$$

◁

Finally, we show that for appropriate parameters to Algorithm 1, the variance of $f(X_{J'},\cdot)$ when $X_{J'}$ is good is small.

▷ Claim 23. If we set $\alpha = \frac{\varepsilon^2}{16}$ in Algorithm 1, then

$$\mathop{\mathbb{E}}_{X_{J'}\in[k+1]^{J'}\text{ good}} \mathrm{Var}\left[f(X_{J'},\cdot)\right] \leq \frac{\varepsilon^2}{8}.$$

Proof. When $X_{J'}$ is good, we have that the marginal gain $\Delta_f(X_{J'}, j, i)$ of adding any $i \in \bar{J}$ to any bucket $j \in [k]$ is at most α. Because of *diminishing marginal gains*, this bounds the marginal gains of $f(X_{J'},\cdot)$ over its entire domain by α. Also, the marginal gains of $f(X_{J'},\cdot)$ are at least $-\alpha$, because if there existed some $Y \geq X_{J'}$, $i \in \bar{J}$ and $j \in [k]$ with $\Delta_f(Y, j, i) < -\alpha$, then by *pairwise monotonicity* we must have for all $j' \neq j$ that $\Delta_f(Y, j', i) \geq \alpha$ as

$$\Delta_f(Y, j, i) + \Delta_f(Y, j', i) \geq 0.$$

But this contradicts the upper bound of α to the marginal gains. Thus, $f(X_{J'},\cdot)$ is α–Lipschitz and we can appeal to Lemma 14 to get

$$\mathrm{Var}[f(x_{J'},\cdot)] \leq 2\alpha\cdot\mathbb{E}[f(x_{J'},\cdot)] \leq \frac{\varepsilon^2}{8}.$$

◁

We are now ready to finish the proof of Theorem 17.

Proof of Theorem 17. We let J' by the output of Algorithm 1 with input f and parameters

$$\delta = \frac{\log(k+1)}{2\log(8\cdot k\cdot |J|\cdot\varepsilon^{-2})}, \qquad \alpha = \frac{\varepsilon^2}{16}.$$

Then the function h defined by Equation (1) is the desired junta function. Indeed, combining (22) and (23), we get

$$\|h - f\|_2^2 \leq \frac{\varepsilon^2}{8} + \frac{\varepsilon^2}{8} = \frac{\varepsilon^2}{4}.$$

To bound the size of J', we use Claim 18 to get

$$|J'| \leq \frac{2}{\alpha\delta} = 32\cdot\frac{1}{\log(1+k)}\cdot\frac{1}{\varepsilon^2}\cdot\log\frac{8\cdot k\cdot|J|}{\varepsilon^2} \leq \frac{32}{\varepsilon^2}\cdot\log\frac{8|J|}{\varepsilon^2}.$$

◀

Using the weak junta approximation, we are able arrive at our final junta approximation.

3.3 Proof of Theorem 2

Let $f : [k+1]^n \to [0,1]$ be a k–submodular function. We will prove a bound of $|J| \leq \frac{4000}{\varepsilon^2} \log \frac{1}{\varepsilon^2}$ for the size of the approximating junta.

Observe that this bound holds trivially for $\varepsilon < n^{-1/2}$, because then we are allowed to choose $J = [n]$. For contradiction, suppose that there is $\varepsilon \in (n^{-1/2}, 1/2)$ for which the statement of Theorem 2 does not hold. Let $\mathcal{E} \subset (n^{-1/2}, 1/2)$ be the set of all ε for which the statement does not hold, and pick an $\varepsilon \in \mathcal{E}$ such that $\varepsilon < 2 \inf \mathcal{E}$. Then, the statement still holds for $\varepsilon_2 = \varepsilon^2 < \frac{1}{2}\varepsilon$.

By the statement of Theorem 2 for ε_2, there is a subset of variables J' of size $|J'| \leq \frac{4000}{\varepsilon^2} \log \frac{1}{\varepsilon} \leq \frac{2^{13}}{\varepsilon^2}$ and a k–submodular function g depending only on J, such that $\|f - g\|_2 \leq \varepsilon_2 \leq \frac{1}{2}\varepsilon$.

Now we apply Theorem 17 to g with parameter ε. Thus, there exists a k–submodular function h such that $\|g - h\|_2 \leq \frac{1}{2}\varepsilon$, and h depends only on a subset of variables $J'' \subseteq J'$, $|J''| \leq \frac{32}{\varepsilon^2} \log \frac{8|J'|}{\varepsilon}$. We have $|J'| \leq \frac{2^{13}}{\varepsilon^2}$, and therefore $|J''| \leq \frac{32}{\varepsilon^2} \log \frac{2^{16}}{\varepsilon^3} \leq \frac{32}{\varepsilon^2} \log \frac{1}{\varepsilon^3}$ (using $\varepsilon \leq \frac{1}{2}$). We conclude that $|J''| \leq \frac{32}{\varepsilon^2} \log \frac{1}{\varepsilon^3} \leq \frac{4000}{\varepsilon^2} \log \frac{1}{\varepsilon}$ as required in Theorem 2. By the triangle inequality, we have $\|f - h\|_2 \leq \|f - g\|_2 + \|g - h\|_2 \leq \frac{1}{2}\varepsilon + \frac{1}{2}\varepsilon = \varepsilon$.

However, this would mean that the statement of Theorem 2 holds for ε as well, which is a contradiction. ◀

4 Towards testing k–submodularity in the ℓ_0-norm

In this section, we develop the local-witness theory for the two constraints that characterize k–submodularity in Hamming distance. For diminishing marginal gains, the witnesses are generalized violated squares. For pairwise monotonicity, the witnesses are violated triangles. We prove that if either constraint is far from being satisfied, then the corresponding witnesses are dense enough to be found by a non-adaptive one-sided tester. The key step in both cases is a repair argument over ideals and filters in the hypergrid.

4.1 Setup

An inductive argument shows that absence of violating squares is equivalent to the diminishing marginal gains property:

▶ **Proposition 24.** *A function $f : [k+1]^n \to \mathbb{R}$ satisfies the diminishing marginal gains property if and only if it has no violating squares.*

Proof. If a function satisfies the diminishing marginal gains property, then it is easy to see that it has no violating squares: take $Y = X_g^j$. We show the converse by considering $X \preceq Y$, $e \notin Y$ and $i \in [k]$. Suppose $e_1, ..., e_\ell$ are the elements of Y not in $X \sqcap Y$ that have $Y(e) \neq k+1$. Then we have that:

$$f(X_e^i) - f(X) \geq f(X_{e,e_1}^{i,Y(e_1)}) - f(X_{e_1}^{Y(e_1)})$$

By induction we can telescopically build the partial partition Y on the right hand side and recover the diminishing marginal gains property. ◀

▶ **Definition 25** (Violating triangle). *Let $X \in [k+1]^n$, $i \neq j \in [k]$, and $e \in X_{k+1}$. We call $\{x, X_e^i, X_e^j\}$ a* triangle. *For a function $f : [k+1]^n \to \mathbb{R}$, this is called an f–*violating triangle *if*

$$\frac{f(X_e^i) + f(X_e^j)}{2} < f(X),$$

that is, it violates the pairwise monotonicity condition.

4.2 Testing diminishing marginal gains

Algorithm 2 A tester for diminishing marginal gains.

Input: Parameters k, n, a query oracle for function $f : [k+1]^n \to \mathbb{R}$, and an error parameter $\varepsilon > 0$.
for $n^2 \cdot (1/\varepsilon)^{\Theta(\sqrt{n \log n})}$ iterations **do**
  sample the following uniformly at random.
    A point $X \in [k+1]^n$ from the hypercube,
    two (possibly same) indices $i, j \in [k]$, and
    two elements $e \neq g \in X_{k+1}$.
  if $\{X, X^i_e, X^j_g, X^{(i,j)}_{(e,g)}\}$ is an f–violating square **then**
    reject.
accept.

We prove the following theorem.

▶ **Theorem 26.** *Algorithm 2 has the following properties.*

1. *If f has* diminishing marginal gains*, then Algorithm 2* **accepts** *with probability* 1.
2. *If f is ε-far from* diminishing marginal gains*, then Algorithm 2* **rejects** *with probability at least* 2/3.
3. *Algorithm 2 makes $q = 4n^2 \cdot (1/\varepsilon)^{\Theta(\sqrt{n \log n})}$ queries.*

Items 1 and 3 are easy to verify – they are direct consequences of Theorem 9 and the description of Algorithm 2, respectively. The rest of this section will be dedicated to proving Item 2.

Intuitively, our proof proceeds as follows. We first show that if a function f has a "low density" of violating squares, then it can be made to have the *diminishing marginal gains* property by changing the values of only a "few" points. By contrapositive, this means that a function that is "far" from *diminishing marginal gains* must have a "high density" of witnesses, which implies the correctness of our tester. To "repair" a function f towards *diminishing marginal gains*, we modify its values over a union of *ideals* and *filters*, defined as follows.

▶ **Definition 27** (Ideal). *Let $X \in [k+1]^n$. The* Ideal *of X is defined as*

$$I(X) := \{Y \mid Y \preceq X\}.$$

▶ **Definition 28** (Filter). *Let $X \in [k+1]^n$. The* Filter *of X is defined as*

$$F(X) := \{Y \mid X \preceq Y\}.$$

We start with the following "repair" lemma for violating squares.

▶ **Lemma 29** (Square repair lemma). *Let $f : [k+1]^n \to \mathbb{R}$ be a function in which $\{X, X^i_e, X^j_g, X^{(i,j)}_{(e,g)}\}$ is an f–violating square. There is a way to change only the values of the ideal $I(X)$ or the filter $F(X^{(i,j)}_{(e,g)})$ to obtain a function f' in which $\{X, X^i_e, X^j_g, X^{(i,j)}_{(e,g)}\}$ is no longer an f'–violating square, and no additional f'–violating squares have been introduced.*

Proof. Because $\{X, X_e^i, X_g^j, X_{(e,g)}^{(i,j)}\}$ is an f–violating square, we have

$$\Delta_X := f(X) + f(X_{(e,g)}^{(i,j)}) - f(X_e^i) - f(X_g^j) > 0.$$

Fix any constant $\Delta \geq \Delta_X$. We obtain f' by subtracting Δ from $f(Y)$ for all $Y \in I(X)$. Clearly, this repairs the f–violating square itself, but we must also argue that it does not introduce any new f'–violating squares.

Let $\{Y, Y_{e'}^{i'}, Y_{g'}^{j'}, Y_{(e',g')}^{(i',j')}\}$ be an f'–violating square that is not f–violating. If $Y \not\preceq X$, then none of the values in the square were modified. So we can assume $Y \preceq X$. We consider the following cases.

1. Neither $Y_{e'}^{i'} \not\preceq X$ nor $Y_{g'}^{j'} \not\preceq X$. Then the points $Y_{e'}^{i'}, Y_{g'}^{j'}, Y_{(e',g')}^{(i',j')}$ are not modified. Thus, the value of this square is

$$(f(Y)-\Delta)+f(Y_{(e',g')}^{(i',j')})-f(Y_{g'}^{j'})-f(Y_{e'}^{i'}) = f(Y)+f(Y_{(e',g')}^{(i',j')})-f(Y_{g'}^{j'})-f(Y_{e'}^{i'})-\Delta \leq -\Delta.$$

2. Only $Y_{e'}^{i'} \not\preceq X$ and $Y_{g'}^{j'} \preceq X$. Then the points $Y_{e'}^{i'}, Y_{(e',g')}^{(i',j')}$ are not modified. Thus, the value of this square is

$$(f(Y)-\Delta)+f(Y_{(e',g')}^{(i',j')})-(f(Y_{g'}^{j'})-\Delta)-f(Y_{e'}^{i'}) = f(Y)+f(Y_{(e',g')}^{(i',j')})-f(Y_{g'}^{j'})-f(Y_{e'}^{i'}) \leq 0.$$

   The case of $Y_{e'}^{i'} \preceq X$ and $Y_{g'}^{j'} \not\preceq X$ is symmetric and identical.

3. Both $Y_{e'}^{i'} \preceq X$ and $Y_{g'}^{j'} \preceq X$. Then $Y_{(e',g')}^{(i',j')} \preceq X$, and all the points are modified. Thus, the value of this square is

$$\begin{aligned}(f(Y) - \Delta) + (f(Y_{(e',g')}^{(i',j')}) - \Delta) - (f(Y_{g'}^{j'}) - \Delta) - (f(Y_{e'}^{i'}) - \Delta)& \\ = f(Y) + f(Y_{(e',g')}^{(i',j')}) - f(Y_{g'}^{j'}) - f(Y_{e'}^{i'}) \leq 0.&\end{aligned}$$

As a result, we have shown that no new violating squares are introduced after changing all the values in $I(X)$, and the original violating square rooted in X is repaired. An identical argument suffices to prove that subtracting Δ from the filter $F(X_{(e,g)}^{(i,j)})$ has the same consequence. ◀

The proof of this lemma also implies the following corollary.

▶ **Corollary 30.** *Let $f : [k+1]^n \to \mathbb{R}$ be a function and $X \in [k+1]^n$ be a domain point. For any constant $\Delta > 0$, the function f' obtained by subtracting Δ from $f(Y)$ for all $Y \in I(X)$ or all $Y \in F(X)$ does not introduce any new f'–violating squares.*

Next, we prove the following density lemma.

▶ **Lemma 31.** *Let $f : [k+1]^n \to \mathbb{R}$ be a function with m violating squares. If $m \leq \varepsilon^{\sqrt{n}\log n} \cdot (k+1)^n$ for $\varepsilon \in (0, e^{-5})$, then f can be made to have* diminishing marginal gains *by changing at most $\varepsilon \cdot (k+1)^n$ values.*

Proof. The proof is a counting argument on the number of points we will need to change by the square repair procedure above. We enumerate all m violating squares and arrange them in order of item-weight (equal to $\sum_{i=1}^{k} |X_i|$) of the bottom point. We split the hypercube evenly based on item-weight. It can be verified that this split occurs at weight $\frac{nk}{k+1}$.

1. For violating squares where the bottom vertex has item-weight $\leq \frac{nk}{k+1}$, we repair them by modifying values in the ideal. Let I denote the bottom vertex in these violating squares. Then the total number of modified points is $|\cup_{X \in I} I(X)|$.

2. For violating squares where the bottom vertex has item-weight $> \frac{nk}{k+1}$, we repair them by modifying values in the filter. Let F denote the top vertex in these violating squares. Then the total number of modified points is $|\cup_{X\in F} F(X)|$.

We first prove the counting argument for Item 1. We estimate this cardinality by combining two different bounds across levels of the hypergrid. Let $L_j := \{X \in [k+1]^n : |X| \le j\}$. Then we can write

$$\left|\bigcup_{X\in I} I(X)\right| = \sum_{j=0}^{\frac{nk}{k+1}} \left|\bigcup_{X\in I} (I(X)\cap L_j)\right|.$$

We can bound these terms in two ways. First, we have

$$\left|\bigcup_{X\in I} (I(X)\cap L_j)\right| \le \sum_{X\in I} |I(X)\cap L_j| = \sum_{X\in I} \binom{|X|}{j} \le |I|\cdot\binom{\frac{nk}{k+1}}{j}.$$

Second, there is the more trivial bound of

$$\left|\bigcup_{X\in I} (I(X)\cap L_j)\right| \le |L_j| = \binom{n}{j}\cdot k^j.$$

This second bound is reminiscent of the binomial distribution $\mathrm{Bin}\left(n, \frac{k}{k+1}\right)$. Combining this with a Chernoff bound, we can get an upper bound on the number of points with item-weight far from the mean. Now we just have to use the correct bound depending on j.

1. If $j \le \frac{nk}{k+1} - \alpha\sqrt{\frac{nk}{k+1}}$ for some constant α, then

$$\begin{aligned}
\sum_{j=0}^{\frac{nk}{k+1}-\alpha\sqrt{\frac{nk}{k+1}}} \binom{n}{j}\cdot k^j &= (k+1)^n \sum_{j=0}^{\frac{nk}{k+1}-\alpha\sqrt{\frac{nk}{k+1}}} \binom{n}{j}\cdot\left(\frac{k}{k+1}\right)^j\left(\frac{1}{k+1}\right)^{n-j} \\
&= (k+1)^n\cdot \Pr\left[Z \le \frac{nk}{k+1} - \alpha\sqrt{\frac{nk}{k+1}}\right] && \left[Z\sim \mathrm{Bin}\left(n,\frac{k}{k+1}\right)\right] \\
&= (k+1)^n\cdot \Pr\left[Z \le (1-\delta)\frac{nk}{k+1}\right] && \left[\text{Letting } \delta = \frac{\alpha}{\sqrt{\frac{nk}{k+1}}}\right] \\
&\le (k+1)^n\cdot e^{-\delta^2 nk/2(k+1)} && \text{[Chernoff bound]} \\
&= (k+1)^n\cdot e^{-\alpha^2/2}.
\end{aligned}$$

2. If $j > \frac{nk}{k+1} - \alpha\sqrt{\frac{nk}{k+1}}$, then

$$\begin{aligned}
\sum_{j=\frac{nk}{k+1}-\alpha\sqrt{\frac{nk}{k+1}}}^{\frac{nk}{k+1}} |I|\cdot\binom{\frac{nk}{k+1}}{j} &= |I|\cdot \sum_{j=0}^{\alpha\sqrt{\frac{nk}{k+1}}} \binom{\frac{nk}{k+1}}{j} \\
&\le |I|\cdot\left(\frac{nk}{k+1}\right)^{\alpha\sqrt{\frac{nk}{k+1}}} \\
&\le |I|\cdot n^{\alpha\sqrt{n}} \\
&= |I|\cdot e^{\alpha\sqrt{n}\ln n}.
\end{aligned}$$

Finally, setting $\alpha = \frac{1}{2}\ln\varepsilon^{-1}$ yields

$$\begin{aligned}
\left|\bigcup_{X\in I} I(X)\right| &= \sum_{j=0}^{\frac{nk}{k+1}} \left|\bigcup_{X\in I}(I(X)\cap L_j)\right| && \text{[Splitting by item-weight]}\\
&= \sum_{j=0}^{\frac{nk}{k+1}-\alpha\sqrt{\frac{nk}{k+1}}} \left|\bigcup_{X\in I}(I(X)\cap L_j)\right| + \sum_{j=\frac{nk}{k+1}-\alpha\sqrt{\frac{nk}{k+1}}}^{\frac{nk}{k+1}} \left|\bigcup_{X\in I}(I(X)\cap L_j)\right| && \text{[Splitting the sum]}\\
&\le \sum_{j=0}^{\frac{nk}{k+1}-\alpha\sqrt{\frac{nk}{k+1}}} \binom{n}{j}\cdot k^j + \sum_{j=\frac{nk}{k+1}-\alpha\sqrt{\frac{nk}{k+1}}}^{\frac{nk}{k+1}} |I|\cdot\binom{\frac{nk}{k+1}}{j} && \text{[Discussed bounds]}\\
&\le (k+1)^n\cdot e^{-\alpha^2/2} + |I|\cdot e^{\alpha\sqrt{n}\ln n} && \text{[From case analysis]}\\
&\le (k+1)^n\cdot e^{-\alpha^2/2} + m\cdot e^{\alpha\sqrt{n}\ln n} && [|I|\le m]\\
&\le (k+1)^n\cdot e^{-\alpha^2/2} + (k+1)^n\cdot\varepsilon^{\sqrt{n}\ln n}\cdot e^{\alpha\sqrt{n}\ln n} && [m\le(k+1)^n\cdot\varepsilon^{\sqrt{n}\ln n}]\\
&= (k+1)^n\cdot\varepsilon^{\frac{1}{8}\ln\varepsilon^{-1}} + (k+1)^n\cdot\varepsilon^{\sqrt{n}\ln n}\cdot\varepsilon^{-\frac{1}{2}\sqrt{n}\ln n} && [\alpha = 1/2\cdot\ln\varepsilon^{-1}]\\
&\le (k+1)^n\cdot\left(\varepsilon^{\frac{1}{8}\ln\varepsilon^{-1}} + \varepsilon^{\frac{1}{2}\sqrt{n}\ln n}\right) && \text{[Simplifying]}\\
&\le \frac{\varepsilon}{2}\cdot(k+1)^n. && [\varepsilon\in(0,e^{-5})]
\end{aligned}$$

This concludes the counting argument for Item 1.

The counting argument for Item 2 is almost identical. We split the filters by level, and for $j \ge \frac{nk}{k+1} + \alpha\sqrt{\frac{nk}{k+1}}$, the Chernoff-inequality based upper bound still holds. For $j < \frac{nk}{k+1} + \alpha\sqrt{\frac{nk}{k+1}}$, we get the following bound.

$$\begin{aligned}
\sum_{j=\frac{nk}{k+1}}^{\frac{nk}{k+1}+\alpha\sqrt{\frac{nk}{k+1}}} \left|\bigcup_{X\in F}(F(X)\cap L_j)\right| &\le \sum_{X\in F}|F(X)\cap L_j|\\
&= \sum_{j=\frac{nk}{k+1}}^{\frac{nk}{k+1}+\alpha\sqrt{\frac{nk}{k+1}}} \sum_{X\in F}\binom{n-|X|}{j-|X|}k^{j-|X|}\\
&\le \sum_{j=\frac{nk}{k+1}}^{\frac{nk}{k+1}+\alpha\sqrt{\frac{nk}{k+1}}} \sum_{X\in F}\binom{\frac{n}{k+1}}{j-|X|}k^{j-|X|}\\
&\le \sum_{j=\frac{nk}{k+1}}^{\frac{nk}{k+1}+\alpha\sqrt{\frac{nk}{k+1}}} \sum_{X\in F}\left(\frac{n}{k+1}\right)^{j-|X|}k^{j-|X|}\\
&\le \sum_{j=\frac{nk}{k+1}}^{\frac{nk}{k+1}+\alpha\sqrt{\frac{nk}{k+1}}} \sum_{X\in F} n^{j-|X|}\\
&\le \sum_{j=\frac{nk}{k+1}}^{\frac{nk}{k+1}+\alpha\sqrt{\frac{nk}{k+1}}} |F|\cdot n^{j-\frac{nk}{k+1}}\\
&\le |F|\cdot n^{\alpha\sqrt{\frac{nk}{k+1}}}\\
&\le |F|\cdot n^{\alpha\sqrt{n}}\\
&= |F|\cdot e^{\alpha\sqrt{n}\ln n}.
\end{aligned}$$

The rest of the counting argument is exactly the same and thus omitted.

In conclusion, we need to modify $\leq \varepsilon \cdot (k+1)^n$ values to make f have *diminishing marginal gains*. ◀

We are now ready to finish the analysis of our tester.

Proof of Theorem 26. We prove each part separately.

1. By Theorem 9, a function with *diminishing marginal gains* does not have any f–violating squares. As a result, the tester will always **accept**.
2. By the contrapositive of Lemma 31, a function that is ε-far from *diminishing marginal gains* has at least $(k+1)^n \cdot \varepsilon^{\sqrt{n}\ln n}$ violating squares. The total number of squares is $\binom{n}{2} \cdot k^2 \cdot (k+1)^{n-2} = \Theta(n^2(k+1)^n)$. Thus, by the witness lemma, $n^2 \cdot (1/\varepsilon)^{\Theta(\sqrt{n \log n})}$ queries suffice to catch an f–violating square with high probability.
3. This follows trivially from the design of the algorithm.

This completes the analysis of Algorithm 2. ◀

4.3 An obstruction to component-wise local repair

The component testers of the previous two sections each rest on a dense witness theorem and a local repair argument. A natural attempt at a tester for full k–submodularity is to apply both component testers in parallel and union-bound the failure probability over the two repair arguments. We now show that this attempt cannot succeed: there exist configurations on which the repair direction prescribed by a triangle witness is forced to oppose the repair direction prescribed by an overlapping square witness on a shared vertex.

For ideals rooted at item-weight $\leq \frac{nk}{k+1}$, the repairs for both witness types proceed downward and can be aligned. The obstruction lives in the upward direction. On filters rooted at item-weight $> \frac{nk}{k+1}$, repairing a violated square requires subtracting $\Delta > 0$ from a vertex, whereas repairing a violated triangle in the same filter requires *adding* $\Delta > 0$ to that vertex. The configuration in Figure 1 realizes both demands simultaneously on the vertices x_2 and x_3.

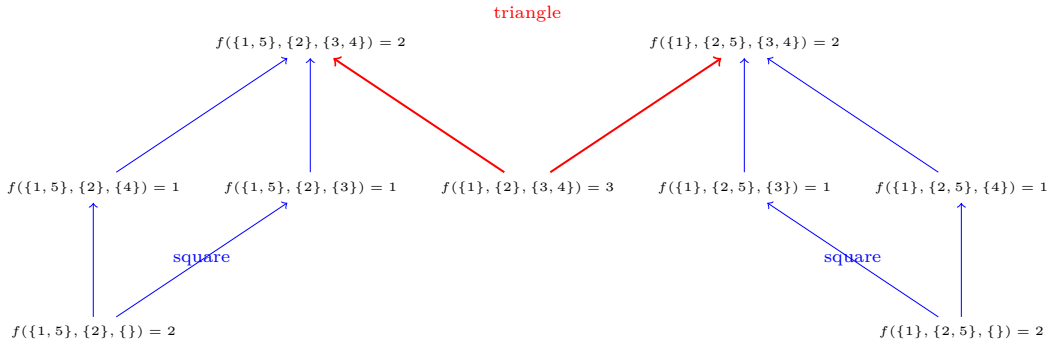


Figure 1 A configuration on a high item-weight filter realizing two incompatible local violations. The red edges form a violating *triangle* on item 5 above x_1: assigning item 5 to part 1 (giving x_2) or to part 2 (giving x_3) both yield marginal -1, summing to $-2 < 0$. The blue edges form two violating *squares* for diminishing marginal gains on items 3 and 4 in part 3, above x_8 (left) and x_9 (right). Repairing the triangle requires raising $f(x_2)$ or $f(x_3)$; repairing the squares requires lowering the same values.

The configuration is not isolated to a small region of the hypergrid. Embedding the same value pattern into any high-item-weight filter on a larger domain produces an instance of the same conflict, so an ideal-only repair strategy that avoids upward filters cannot route around it without losing the balance argument that makes the component testers work.

▶ **Observation 32** (Component-wise repair is insufficient). *For every $k \geq 2$ there exist functions $f : [k+1]^n \to \mathbb{R}$ and high item-weight filters on which a violated triangle witness and an overlapping violated square witness prescribe local repairs of opposite sign on a common vertex. In particular, the standard component-wise repair argument – repair triangles and squares independently and combine via union bound – does not yield a tester for k-submodularity.*

Observation 32 does not rule out a Hamming-distance tester for k-submodularity; it shows only that obtaining one will require either a richer witness vocabulary that simultaneously certifies both component constraints, or a non-local repair scheme that mediates between square and triangle repairs across the hypergrid.

References

1 Eric Blais and Abhinav Bommireddi. Testing submodularity and other properties of valuation functions. *arXiv preprint arXiv:1611.07879*, 2016. arXiv:1611.07879.

2 Stéphane Boucheron, Gábor Lugosi, and Olivier Bousquet. Concentration inequalities. In *Summer school on machine learning*, pages 208–240. Springer, 2003. doi:10.1007/978-3-540-28650-9_9.

3 Deeparnab Chakrabarty and Zhiyi Huang. Recognizing coverage functions. *SIAM Journal on Discrete Mathematics*, 29(3):1585–1599, 2015. doi:10.1137/140964072.

4 Alina Ene and Huy Nguyen. Streaming algorithm for monotone k-submodular maximization with cardinality constraints. In *International Conference on Machine Learning*, pages 5944–5967. PMLR, 2022. URL: https://proceedings.mlr.press/v162/ene22a.html.

5 Jon Feldman, Monika Henzinger, Nitish Korula, Vahab S Mirrokni, and Cliff Stein. Online stochastic packing applied to display ad allocation. In *European Symposium on Algorithms*, pages 182–194. Springer, 2010. doi:10.1007/978-3-642-15775-2_16.

6 Vitaly Feldman and Jan Vondrák. Optimal bounds on approximation of submodular and xos functions by juntas. *SIAM Journal on Computing*, 45(3):1129–1170, 2016. doi:10.1137/140958207.

7 Michel X Goemans and Jan Vondrák. Covering minimum spanning trees of random subgraphs. *Random Structures & Algorithms*, 29(3):257–276, 2006. doi:10.1002/RSA.20115.

8 Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998. doi:10.1145/285055.285060.

9 Themistoklis Haris and Diptaksho Palit. Testing k-submodularity. *CoRR*, abs/2606.30433, 2026. doi:10.48550/arXiv.2606.30433.

10 Pooya Hatami and Jan Vondrák. Testing real-valued modularity and submodularity, 2016. URL: https://www.ias.edu/sites/default/files/math/csdm/15-16/HatamiVondrak.pdf.

11 Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM J. Comput.*, 22(6):1331–1348, 1993. doi:10.1137/0222080.

12 Naoto Ohsaka and Yuichi Yoshida. Monotone k-submodular function maximization with size constraints. *Advances in Neural Information Processing Systems*, 28, 2015.

13 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. On testing convexity and submodularity. *SIAM Journal on Computing*, 32(5):1158–1184, 2003. doi:10.1137/S0097539702414026.

14 Sofya Raskhodnikova. CS 599: Sublinear Algorithms (Spring 2025). https://cs-people.bu.edu/sofya/sublinear-course/, 2025. Retrieved on 10 October, 2025. URL: https://cs-people.bu.edu/sofya/sublinear-course/.

15 Sofya Raskhodnikova and Grigory Yaroslavtsev. Learning pseudo-boolean k-dnf and submodular functions. In Sanjeev Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1356–1368. SIAM, 2013. doi:10.1137/1.9781611973105.98.

16 Comandur Seshadhri and Jan Vondrák. Is submodularity testable? *Algorithmica*, 69(1):1–25, 2014. doi:10.1007/S00453-012-9719-2.

17 Justin Ward and Stanislav Živnỳ. Maximizing k-submodular functions and beyond. *ACM Transactions on Algorithms (TALG)*, 12(4):1–26, 2016. doi:10.1145/2850419.

Symmetric Distributions from Shallow Circuits

Daniel M. Kane
UC San Diego, La Jolla, CA, USA

Anthony Ostuni
UC San Diego, La Jolla, CA, USA

Kewen Wu
Caltech, Pasadena, CA, USA

Abstract

We characterize the symmetric distributions that can be (approximately) generated by shallow Boolean circuits. More precisely, let $f\colon \{0,1\}^m \to \{0,1\}^n$ be a Boolean function where each output bit depends on at most d input bits. Suppose the output distribution of f evaluated on uniformly random input bits is close in total variation distance to a symmetric distribution $\mathcal{D}$ over $\{0,1\}^n$. Then $\mathcal{D}$ must be close to a mixture of the uniform distribution over n-bit strings of even Hamming weight, the uniform distribution over n-bit strings of odd Hamming weight, and γ-biased product distributions for γ an integer multiple of 2^{-d}. Moreover, the mixing weights are determined by low-degree, sparse $\mathbb{F}_2$-polynomials. This extends the previous classification for generating symmetric distributions that are also uniform over their support.

2012 ACM Subject Classification Theory of computation → Circuit complexity

Keywords and phrases Sampling, distributions, locality, circuit complexity

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.53

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2511.14127` [32]

Funding *Daniel M. Kane*: Supported by NSF Medium Award CCF-2107547 and NSF CAREER Award CCF-1553288.

Acknowledgements We thank Farzan Byramji and anonymous reviewers for helpful comments on an earlier draft of this paper.

1 Introduction

One of the most celebrated results in complexity theory is that AC^0 circuits[1] require an exponential number of gates to compute the parity function [22, 1, 45, 26, 27]. Surprisingly, the weaker circuit class NC^0 suffices to perform a very similar task. In particular, mapping uniformly random bits $(x_1, x_2, \ldots, x_n)$ to $(x_1 \oplus x_2, x_2 \oplus x_3, \ldots, x_{n-1} \oplus x_n, x_n \oplus x_1)$ produces the uniform distribution over n-bit strings of even parity, or equivalently, over input-output pairs $(x, \mathrm{PARITY}(x))$ [5, 10]. This observation begs the question: what computational resources are required to (approximately) *generate* specific distributions, as opposed to the traditional task of *computing* specific functions?

A fundamental question in its own right, the complexity of sampling from distributions also has numerous applications. For example, results on the hardness of sampling can be translated to data structure lower bounds [40, 35, 6, 43, 11, 44, 47, 30, 2], provide input-independent quantum-classical separations [7, 44, 30, 24], and are key components

[1] Recall that these are (families of) Boolean circuits of constant depth and unbounded fan-in gates. They may be contrasted with NC^0 circuits, which are also of constant depth, but have bounded fan-in gates.

© Daniel M. Kane, Anthony Ostuni, and Kewen Wu;
licensed under Creative Commons License CC-BY 4.0

to the construction of explicit codes [37]. Moreover, techniques and intuition developed in this setting have successfully been applied to pseudorandom generators [40, 35, 6] and extractors [41, 17, 42, 13, 14].

While the general problem was considered in early work (see, e.g., [29]), a focus on generating distributions via shallow circuits was advocated for more recently by Viola [40]. Since then, the field has seen a number of exciting developments (see, e.g., the recent works [21, 44, 30, 37, 31, 2] and references therein). One notable takeaway from prior works is that NC^0 circuits can sample very few uniform symmetric distributions (i.e., uniform distributions over a symmetric support), even allowing for small errors. In particular, the line of work [40, 21, 44, 30, 31] recently culminated in the following classification result, which confirmed a conjecture of Filmus, Leigh, Riazanov, and Sokolov [21]. For a function $f\colon \{0,1\}^m \to \{0,1\}^n$, let $f(\mathcal{U}^m)$ be the distribution resulting from applying f to $x \sim \mathcal{U}^m$, the uniform distribution over $\{0,1\}^m$.

▶ **Theorem 1** ([31]). *Let $\varepsilon \in [0,1]$ be arbitrary. Assume $f\colon \{0,1\}^m \to \{0,1\}^n$ is computable by an NC^0 circuit of constant depth and $f(\mathcal{U}^m)$ is ε-close in total variation distance to a uniform symmetric distribution where n is sufficiently large. Then $f(\mathcal{U}^m)$ is $O(\varepsilon)$-close to one of the following six special uniform symmetric distributions:*

- *Point distribution on 0^n,*
- *Point distribution on 1^n,*
- *Uniform distribution over $\{0^n, 1^n\}$,*
- *Uniform distribution over n-bit strings with even Hamming weights,*
- *Uniform distribution over n-bit strings with odd Hamming weights,*
- *Uniform distribution over all n-bit strings.*

We emphasize that Theorem 1 works with *uniform* symmetric distributions, which does not capture, for example, the $(1/4)$-biased product distribution that is symmetric and easily sampleable by NC^0 circuits.

1.1 Our Result

In this paper, we take another step toward understanding the sampleability of distributions by extending Theorem 1 to handle *arbitrary* symmetric distributions. Let the *locality* of a Boolean function be the largest number of input bits that any output bit depends on. Note that NC^0 is precisely the class of sequences of functions with bounded locality.

▶ **Theorem 2** (Informal version of [32, Theorem 4.1]). *Let $d \geq 0$ be an integer. For any $\varepsilon \in (0,1]$, there exists some δ such that $\delta \to 0$ as $\varepsilon \to 0$ and the following holds.*

Suppose $f\colon \{0,1\}^m \to \{0,1\}^n$ is a d-local function and $f(\mathcal{U}^m)$ is ε-close in total variation distance to a symmetric distribution where n is sufficiently large in terms of d and ε. Then $f(\mathcal{U}^m)$ is δ-close to some mixture of

1. *The uniform distribution over n-bit strings of even Hamming weight,*
2. *The uniform distribution over n-bit strings of odd Hamming weight, and*
3. *γ-biased product distributions on n bits for γ an integer multiple of 2^{-d}.*

Moreover, the mixing weights are determined by degree-$O_d(1)$ $\mathbb{F}_2$-polynomials with $O_d(n)$ monomials.

The "moreover" conclusion implies this mixture can be exactly produced by $O_d(1)$-local functions (see [32, Remark 4.2]).

Learning Structured Distributions

As mentioned above, the study of sampling is rife with applications to other areas of theoretical computer science. Continuing this trend, we highlight one consequence of Theorem 2 to learning theory.

The reconstruction of an unknown probability density function based on observed data is a fundamental problem in both statistics and computer science. The typical setting is the PAC-learning model [38, 9, 33]: given access to independent samples of an unknown distribution $\mathcal{D}$, the goal is to output a hypothesis distribution $\mathcal{D}'$ close to $\mathcal{D}$. Much research has been carried out for Gaussian mixtures [16], log-concave distributions [20], monotone distributions [8], sums of independent integer random variables [15], junta distributions [3], mixtures of structured distributions [34], and more. The interested reader may wish to consult the survey [18] by Diakonikolas for additional background and references.

A black-box use of Theorem 2 in this setting is to learn symmetric distributions that are locally sampleable. Let $\mathcal{D}$ be a symmetric distribution over $\{0,1\}^n$. Assume $\mathcal{D}$ is produced by a d-local function, i.e., $\mathcal{D} = f(\mathcal{U}^m)$ for some d-local function $f\colon \{0,1\}^m \to \{0,1\}^n$. Then for any $\varepsilon > 0$ with n sufficiently large in terms of d and ε, Theorem 2 and the standard cover method [46] (see also [18, Theorem 1.5.1]) imply that we can efficiently learn $\mathcal{D}$ up to ε-error in total variation distance with $O_d(1/\varepsilon^2)$ samples with high probability. We believe this should hold for all $\varepsilon > 0$. Indeed, in Section 1.2 we propose Conjecture 4 for an exact classification of locally sampleable symmetric distributions, which, if true, would imply the desired learning theoretic result.

1.2 Open Problems

Recall that Theorem 2 does not work for the case of $\varepsilon = 0$, i.e., *exact* sampling. Indeed, we cannot conclude mixtures of the form given by Theorem 2 are the only distributions that can be exactly sampled by NC^0 circuits. We identify the following example, which illustrates that this is not simply a weakness in our analysis, but rather there are other symmetric distributions that can be sampled exactly.

▶ **Example 3.** Let $\mathcal{U}^n_{1/4}$ be the $(1/4)$-biased product distribution over $\{0,1\}^n$. Additionally, let $\texttt{evens}$ and $\texttt{odds}$ denote the uniform distribution over n-bit strings of even Hamming weight and odd Hamming weight, respectively. The distribution[2]

$$\mathcal{P} = \mathcal{U}^n_{1/4} + 2^{-n-1}\texttt{evens} - 2^{-n-1}\texttt{odds}$$

is not of the form given by Theorem 2, yet it can be sampled exactly by a bitwise AND of $\texttt{evens}$ and the uniform distribution over $\{0,1\}^n$, which is 3-local. The full details can be found in [32, Appendix A].

Additionally, we suspect that, similarly to the uniform symmetric case [31], one can take the upper bound in Theorem 2 to be linear in $\|f(\mathcal{U}^m) - \mathcal{D}\|_{\mathsf{TV}}$ without any dependency on d. Combining with the previous discussion, we conjecture the following strengthening of Theorem 2 holds. Observe that Conjecture 4 captures Example 3.

▶ **Conjecture 4.** *For every $d \in \mathbb{N}$, $\varepsilon \in (0,1)$, and n large enough in terms of d, if $f\colon \{0,1\}^m \to \{0,1\}^n$ is a d-local function and $f(\mathcal{U}^m)$ is ε-close in total variation distance to a symmetric distribution, then $f(\mathcal{U}^m)$ is $O(\varepsilon)$-close to some mixture*

$$\mathcal{M} = \sum_i \alpha_i \cdot g_i(\mathcal{D}^{(i)}_1, \ldots, \mathcal{D}^{(i)}_d),$$

[2] More formally, $\mathcal{P}$ is the distribution that assigns x probability $\mathcal{U}^n_{1/4}(x) + 2^{-n-1}\texttt{evens}(x) - 2^{-n-1}\texttt{odds}(x)$.

where each g_i is a bitwise function and each $\mathcal{D}_j^{(i)}$ is either the uniform distribution over n-bit strings of even Hamming weight or over n-bit strings of odd Hamming weight. Moreover, the mixing weights are determined by degree-$O_d(1)$ $\mathbb{F}_2$-polynomials with $O_d(n)$ monomials, as in Theorem 2.

Note the bitwise condition on the above g_i's guarantees the output distribution is symmetric.

Paper Organization

We provide an overview of the proof of Theorem 2 in Section 2, as well as a brief comparison of our techniques to those of prior literature. Our full results and proofs can be found in the arXiv version of the paper [32].

2 Proof Overview

In this section, we sketch the proof of Theorem 2 before discussing how the details compare to prior work.

2.1 Proof Overview of Theorem 2

We begin with a useful observation from [31]: the total variation distance between the distribution $f(\mathcal{U}^m)$ and any symmetric distribution $\mathcal{P}$ over $\{0,1\}^n$ is, up to constant factors, equal to the distance between the corresponding Hamming weight distributions of $f(\mathcal{U}^m)$ and $\mathcal{P}$ plus the distance between $f(\mathcal{U}^m)$ and its symmetrization (i.e., the distribution resulting from randomly permuting the coordinates of a string $x \sim f(\mathcal{U}^m)$). Expressed symbolically, we have

$$\|f(\mathcal{U}^m) - \mathcal{P}\|_{\mathsf{TV}} = \Theta(\||f(\mathcal{U}^m)| - |\mathcal{P}|\|_{\mathsf{TV}} + \|f(\mathcal{U}^m) - f(\mathcal{U}^m)_{\mathrm{sym}}\|_{\mathsf{TV}}). \tag{1}$$

The proof is straightforward, and it essentially follows from several applications of the triangle inequality which show that the two ways $f(\mathcal{U}^m)$ can be far from $\mathcal{P}$ are a weight mismatch or $f(\mathcal{U}^m)$ being far from symmetric itself (see [32, Lemma 3.5]).

By assumption, we know $f(\mathcal{U}^m)$ is ε-close to a symmetric distribution $\mathcal{D}$. Thus, Equation (1) implies $\|f(\mathcal{U}^m) - f(\mathcal{U}^m)_{\mathrm{sym}}\|_{\mathsf{TV}} = O(\varepsilon)$. Hence, it suffices to prove a weaker version of Theorem 2 that only compares the weight distributions ([32, Lemma 4.26]). More precisely, we want to show that for some δ tending to 0 with ε, the Hamming weight distribution of $f(\mathcal{U}^m)$, denoted $|f(\mathcal{U}^m)|$, is δ-close to a mixture $\mathcal{M}$ of

1. The binomial distribution $\mathsf{Bin}(n, \gamma)$ with n trials and success probability γ for γ an integer multiple of 2^{-d},
2. The binomial distribution $\mathsf{Bin}(n, 1/2)$ conditioned on the outcome being even, and
3. The binomial distribution $\mathsf{Bin}(n, 1/2)$ conditioned on the outcome being odd.

Moreover, we want the mixing weights to be determined by low-degree $\mathbb{F}_2$-polynomials with few monomials. (It will later become clear what "determined" means in this context.) We first prove a weaker version of this result that does not include mixing weight information ([32, Lemma 4.24]). Afterwards, we will discuss how to obtain the desired control over the weights. Before proceeding to the details, we first give a brief, high-level overview of the four main steps of the proof and how they fit together.

In the first step, we show that any fixing of the values of input bits which affect many output bits results in the bias of the output weight distribution of $f(\mathcal{U}^m)$ concentrating around a fixed dyadic rational[3] γ multiple of n. This allows us to represent the weight distribution $|f(\mathcal{U}^m)|$ as a mixture of distributions produced by the restricted functions. We then argue in step two that after grouping the parts of the mixture which concentrate around the same γ, each group assigns roughly the same amount of mass to any contiguous interval as the binomial distribution with success probability γ. The third step is to prove a continuity result for each grouped part of the mixture; namely, that the mass assigned to some weight w and to $w + \Delta$ for a small integer Δ are roughly equal. (There is a slight subtlety here in the case of $\gamma = 1/2$, but we will defer its discussion to later in the proof overview.) This allows us to conclude in the fourth and final step that most output weights are assigned comparable mass by the part of the mixture of $|f(\mathcal{U}^m)|$ concentrated around γ and the corresponding binomial distribution. In particular, our weight distribution is close in total variation distance to a mixture of binomial distributions, which is what we wanted to show.

2.1.1 Step 1: Removing Large Influences

In an extremely ideal setting, we might wish that all of f's output bits are roughly independent with bias around $\gamma = a/2^d$ for some integer $0 \le a \le 2^d$; in this case the output weight would resemble the binomial distribution $\mathsf{Bin}(n, \gamma)$. Of course, this is far too much to assume. In actuality, it may be that one specific input bit affects the value of *every* output bit, so none of them are independent.

To progress toward this dream scenario, we follow in the footsteps of many prior works (e.g., [40, 35, 6, 43, 44, 21, 30, 31, 24]) by strategically conditioning on certain input bits to express the distribution as a mixture of more structured sub-distributions. For example, suppose some input bit b affects many output bits. Since b takes value 0 and 1 with equal probability, we may express the distribution $f(\mathcal{U}^m)$ as the mixture

$$f(\mathcal{U}^m) = \frac{1}{2} \cdot f\left(\mathcal{U}^m \mid b = 0\right) + \frac{1}{2} \cdot f\left(\mathcal{U}^m \mid b = 1\right).$$

Observe that the large influence of b is no longer a problem in either of the restricted distributions, as its value is fixed.

In particular, we will condition on all "high degree" input bits that affect more than n/A output bits for some A to be chosen later. The goal is now to argue that the function resulting from each conditioning produces a distribution which is roughly of the form we originally sought. Note that this will not depend on the actual values that the input bits are set to in the conditioning.

We start with a result from [30]: let $\mathcal{D}_q$ be the uniform distribution over n-bit strings of Hamming weight q. If q/n is far from every integer multiple of 2^{-d}, then any d-local function must produce a distribution far (in total variation distance) from $\mathcal{D}_q$. This essentially follows from the fact that the input bits determining any fixed output bit can only be set in 2^d equally likely ways. Observe that symmetric distributions are simply mixtures of $\mathcal{D}_q$ for different q's. Thus we can show ([32, Lemma 4.8]) that if $f(\mathcal{U}^m)$ is close to a symmetric distribution $\mathcal{D}$, then for a typical $x \sim \mathcal{U}^m$, the normalized output weight $|f(x)|/n$ has distance at most $n^{-1/(800d)}$ from an integer multiple of 2^{-d}. Note that the closest multiple may be different for different inputs; however, after each conditioning $\rho \in \{0,1\}^S$ on the set of high degree input

[3] Recall a *dyadic rational* is a number that can be expressed as a fraction whose denominator is a power of two.

bits $S \subseteq [m] := \{1, 2, \ldots, m\}$, we can strengthen this result ([32, Lemma 4.9]) to say that for a typical $x \sim \mathcal{U}^{[m]\setminus S}$, the normalized output weight $|f(x, \rho)|/n$ is close to the *same* integer multiple of 2^{-d}. Here, we will only need "high degree" to correspond to affecting more than $n/O_d(1)$ many output bits (i.e., can take $A = O_d(1)$), although later we will obtain other constraints on how we must set A. Henceforth, we will shorthand $f(x, \rho)$ by $f_\rho(x)$ for clarity.

At a high level, the proof uses the second-moment method. By conditioning on the high degree input bits, we ensure that the variance of the resulting distribution is small, and thus we have concentration around a particular weight. Moreover, this weight must be close to an integer multiple of $n/2^d$, or else our previous insights imply $f(\mathcal{U}^m)$ cannot be close to $\mathcal{D}$. To be slightly more precise, a direct second-moment argument by itself is insufficient to rule out the case that some non-negligible fraction of the time the output weight is close to some other multiple of $n/2^d$. However, one can show ([32, Claim 4.14]) that if this occurs, $f_\rho(\mathcal{U}^{[m]\setminus S})$ would also have to assign decent probability mass to weights between these integer multiples, which would again imply $f(\mathcal{U}^m)$ cannot be close to $\mathcal{D}$.

Finally, we combine these deductions to prove that for each conditioning $\rho \in \{0, 1\}^S$ of the bits in S, there exists a set $T := T_\rho \subseteq [n]$ of size $|T| \leq O_{d,k}(1)$ such that every k-tuple of output bits in $[n] \setminus T$ has marginal distribution $\mathcal{U}^k_{\gamma_\rho}$, the γ_ρ-biased product distribution over $\{0, 1\}^k$, for γ_ρ an integer multiple of 2^{-d} (see [32, Proposition 4.5]). Here, k is some parameter at most $O_d(\log(1/\varepsilon))$. The proof operates by constructing a degree-k multilinear polynomial $P_i \colon \{0, 1\}^n \to \{0, 1\}$ for each k-tuple with the "wrong" distribution, where the expectation of P_i over $f_\rho(\mathcal{U}^{[m]\setminus S})$ is larger than over the γ-biased product distribution $\mathcal{U}^n_\gamma$ by at least an additive 2^{-kd} term. Note this follows from using our locality bound to view $P_i(f_\rho(\mathcal{U}^{[m]\setminus S}))$ as a polynomial of degree kd in the input bits. Summing the polynomials together into $P = \sum_i P_i$ magnifies the difference in expectations, and if the number of terms (i.e., number of bad k-tuples) is too large, we end up contradicting the assumption that $f(\mathcal{U}^m)$ is close to a symmetric distribution.

2.1.2 Step 2: Kolmogorov Distance

We now group the restricted functions according to their biases, defining $F_\gamma(\mathcal{U}^{[m]\setminus S}) = \mathbb{E}_{\rho:\gamma_\rho=\gamma}[f_\rho(\mathcal{U}^{[m]\setminus S})]$ for each γ. In this second step, we aim to show that every contiguous interval of output weights is assigned roughly the same amount of mass by $|F_\gamma(\mathcal{U}^{[m]\setminus S})|$ and $\mathsf{Bin}(n, \gamma)$. Since this difference in probability mass is convex over mixtures, it suffices to consider the individual distributions $|f_\rho(\mathcal{U}^{[m]\setminus S})|$. Moreover, it is enough to provide a bound on the *Kolmogorov distance*

$$\max_t \left| \Pr\left[|f_\rho(\mathcal{U}^{[m]\setminus S})| \geq t\right] - \Pr\left[\mathsf{Bin}(n, \gamma) \geq t\right] \right|.$$

In a sentence, the bound follows from combining the k-wise independence of most output coordinates with the fact that T is too small to have much of an effect. In the case of $\gamma = 1/2$ and $|T| = 0$, Diakonikolas, Gopalan, Jaiswal, Servedio, and Viola [19] gave an upper bound of roughly $1/\sqrt{k}$ using techniques from approximation theory. (In fact, their result holds for arbitrary threshold functions.) This was generalized by Gopalan, O'Donnell, Wu, and Zuckerman [23] to include the case of arbitrary $\gamma \in (0, 1)$. In our case, $|T|$ will likely not be 0, but it is small enough that a similar result ([32, Proposition 4.16]) still holds. In the edge cases of $\gamma \in \{0, 1\}$, we can no longer use [19, 23], nor would the size of T be negligible even if we could, but these special cases can be addressed later via simple arguments (see, e.g., the proof of [32, Lemma 4.23]).

For reasons that will become apparent in the subsequent step, we will also need a comparable bound in the case of $\gamma = 1/2$, even accounting for the parity of the output weight. That is, we wish to show

$$\Pr\left[|f_\rho(\mathcal{U}^{[m]\setminus S})| \geq t \text{ and } |f_\rho(\mathcal{U}^{[m]\setminus S})| \text{ is even}\right] \approx \Pr\left[\mathsf{Bin}(n,1/2) \geq t\right] \cdot \Pr\left[|f_\rho(\mathcal{U}^{[m]\setminus S})| \text{ is even}\right].$$

Our analysis here is similar to the previous case, but we now crucially rely on [12, Theorem 3.1]. In our context, it implies that there is a small set of input bits $R \subseteq [m] \setminus S$ such that re-randomizing over R typically re-randomizes the parity of f_ρ's output weight. Moreover, since R is small and we have already conditioned on input bits of large degree, the vast majority of output bits are unaffected by R. Thus, we can compare $\Pr\left[|f_\rho(\mathcal{U}^{[m]\setminus S})| \geq t\right]$ to $\Pr\left[\mathsf{Bin}(n,1/2) \geq t\right]$ using these unaffected output bits as a proxy for the entirety of the output bits. We briefly note that for the error bounds on the "parity Kolmogorov distance" to be meaningful given our choice of k, we need to take A (defined in our earlier high degree threshold) to be $O_{d,\varepsilon}(1)$ – larger than is necessary for Step 1.

2.1.3 Step 3: Approximate Continuity

Our third step is to argue that for each bias γ, the distribution $F_\gamma(\mathcal{U}^{[m]\setminus S})$ can be expressed as a mixture $\lambda \cdot E_\gamma + (1-\lambda) \cdot W_\gamma$, where λ is small and W_γ assigns similar probability mass to similar weights. Once again, it suffices to analyze the distributions produced by the individual restricted functions $f_\rho(\mathcal{U}^{[m]\setminus S})$. Here, we show that $f_\rho(\mathcal{U}^{[m]\setminus S})$ can be written as a mixture of distributions E and W, where E is extremely far from every symmetric distribution supported on strings of Hamming weight $\gamma n \pm n^{2/3}$, and the weight distribution of W satisfies a certain continuity property (see [32, Proposition 4.21]). More specifically, for positive integers w and Δ, the probability that W's output weight is w differs by no more than $O_d(\Delta/n)$ from the probability that W's output weight is $w + \Delta$. Since $f(\mathcal{U}^m)$ is close to $\mathcal{D}$, one can show that $F_\gamma(\mathcal{U}^{[m]\setminus S})$ is close to $\mathcal{D}$ conditioned on the output weight being $\gamma n \pm n^{2/3}$ (see [32, Claim 4.25]). Thus, proving the above result for $f_\rho(\mathcal{U}^{[m]\setminus S})$ implies minimal mass will typically be assigned to the E part of the mixtures, and so λ (in the mixture defining $F_\gamma(\mathcal{U}^{[m]\setminus S})$) will be small, as desired.

In our analysis, we will require a structural result about hypergraphs from [30]. Observe that we can associate any function $g\colon \{0,1\}^m \to \{0,1\}^n$ to a hypergraph on the vertex set $[n]$ with an edge for each input bit b containing all of the output bits that depend on b. In the case that g is d-local, this hypergraph has maximum degree at most d. We follow standard hypergraph terminology in defining the *neighborhood* of a vertex v to be the set of vertices sharing an edge with v. We additionally call two neighborhoods N_1, N_2 *connected* if there exist two adjacent (i.e., contained in the same edge) vertices $v_1 \in N_1, v_2 \in N_2$.

By [30, Corollary 4.11], we can find a collection of $r = \Omega_d(n)$ non-connected neighborhoods in $\mathcal{H}$ of size $O_d(1)$ by only removing $O_d(n)$ (with a small implicit constant) edges. Translating the result to f_ρ, we find that there exists a not-too-large set $B \subseteq [m]\setminus S$ of input bits such that any conditioning $\sigma \in \{0,1\}^B$ yields a sub-function $f_{\rho,\sigma}\colon \{0,1\}^{[m]\setminus(S\cup B)} \to \{0,1\}^n$ with many small, pairwise independent collections of output bits. We will define E and W according to the behavior of these neighborhoods, similarly to the arguments in [30, 31, 24].

Suppose for at least $r/2$ of the r non-connected neighborhoods $N_1, \ldots, N_r$, we have that $f_{\rho,\sigma}(\mathcal{U}^{[m]\setminus(S\cup B)})$ restricted to the neighborhood N_i has a marginal distribution which differs from the γ-biased product distribution over N_i, denoted $\mathcal{U}_\gamma^{N_i}$. Since $\mathcal{U}_\gamma^{N_i}$ is pointwise

close to the uniform distribution over strings of Hamming weight around γn, the marginal distributions of these neighborhoods (in $f_{\rho,\sigma}$) must also differ from the marginal distributions of the symmetric distribution $\mathcal{D}$. We can then accumulate these errors via concentration bounds to show that $f_{\rho,\sigma}(\mathcal{U}^{[m]\setminus(S\cup B)})$ is far from $\mathcal{D}$ (see [32, Lemma 3.3]).

We now set E to be the mixture over all conditions of the bits in B where most resulting neighborhoods differ from the corresponding γ-biased product distribution, and set W to be the mixture over the remaining conditionings. Since each sub-distribution in the mixture E is far from $\mathcal{D}$ by the previous paragraph, a union bound argument implies E, itself, must also be far from $\mathcal{D}$ (see [32, Lemma 3.4]). It remains to show the continuity property for W.

Suppose the r non-connected neighborhoods are generated by $v_1, v_2, \ldots, v_r \in [n]$ in the sense that all bits in the i-th neighborhood N_i are either v_i or in an edge that also contains v_i. We further condition on all input bits that do not affect any of $v_1, \ldots, v_r$, so that the value of every output bit outside of $N_1 \cup \cdots \cup N_r$ is fixed. In this way, the output weight of $f_{\rho,\sigma}(\mathcal{U}^{[m]\setminus(S\cup B)})$ becomes a fixed integer (corresponding to the fixed bits outside of the neighborhoods) plus the sum of the neighborhoods' output weights. Since we are in the case where most neighborhoods are extremely structured, we can show that for each $2 \leq \ell \leq t$, the output weight distribution modulo ℓ of N_i is not constant for a constant fraction of the N_i's with high probability (see [32, Claim 4.22]). Finally, we can obtain our desired continuity result through known density comparison theorems for sums of independent, non-constant integer random variables, such as [31, Theorem A.1], which relies on classical anticoncentration tools.

There is one important exception to the above analysis: the case of $\ell = 2$ and $\gamma = 1/2$. Here, we are not guaranteed to typically get many non-connected neighborhoods with variable output weight modulo 2 (even if most of them have the "correct" marginal distribution). To better understand where our reasoning breaks down, consider one of the most surprising examples in this area of work. Define $h\colon \{0,1\}^n \to \{0,1\}^n$ to be

$$h(x_1, \ldots, x_n) = (x_1 \oplus x_2, x_2 \oplus x_3, \ldots, x_{n-1} \oplus x_n, x_n \oplus x_1),$$

so that $h(\mathcal{U}^n)$ is the uniform distribution over n-bit strings of even Hamming weight. Observe that h is extremely simple, only requiring two bits of locality. Additionally, the marginal distribution onto any $k \leq n-1$ coordinates is exactly the product distribution $\mathcal{U}_{1/2}^k$.

Let us focus on a particular output bit $y_i = x_i \oplus x_{i+1}$. (For simplicity, assume i is not too close to 1 or n.) Its neighborhood is $y_{i-1} = x_{i-1} \oplus x_i$, y_i, and $y_{i+1} = x_{i+1} \oplus x_{i+2}$, so its Hamming weight modulo 2 is

$$(x_{i-1} \oplus x_i) \oplus (x_i \oplus x_{i+1}) \oplus (x_{i+1} \oplus x_{i+2}) = x_{i-1} \oplus x_{i+2}.$$

If we follow our earlier analysis and fix the value of the input bits that do not affect y_i (i.e., x_{i-1} and x_{i+2}), the neighborhood's Hamming weight modulo two is fixed, regardless of the values of x_i and x_{i+1}.

At a high level, the reason for this exceptional case is that it is the only setting of ℓ and γ for which the binomial distribution $\mathsf{Bin}(t,\gamma)$ mod ℓ can equal $\mathsf{Bin}(t-1,\gamma)$ mod ℓ. In other words, the weight distribution modulo ℓ of $\mathcal{U}_\gamma^k$ conditioned on the first bit being 0 is different than that distribution conditioned on the first bit being 1, unless $\ell = 2$ and $\gamma = 1/2$. This difference means that in the case of $\ell = 2$ and $\gamma = 1/2$, we can only derive a continuity result for weights that are an even distance apart. Hence, we must be cognizant of the output weight's parity throughout much of our analysis, which explains the required parity version of our Kolmogorov distance result mentioned earlier in the proof overview.

2.1.4 Step 4: Putting It Together

At this point, all that remains is to combine the pieces. Recall we have already conditioned on high degree (i.e., larger than $n/O_{d,\varepsilon}(1)$) input bits to find sub-functions $\{f_\rho\}_\rho$, each producing a distribution with some approximate bias γ_ρ. Additionally, the mixtures $F_\gamma(\mathcal{U}^{[m]\setminus S})$ obtained by grouping the sub-functions around their biases each satisfy a Kolmogorov distance bound and approximate continuity.

We partition $\{0, 1, \ldots, n\}$ into consecutive intervals of length $c\sqrt{n}$ for some small $c > 0$, and restrict our attention to the $O(\log(1/\alpha))$ of them that contain all but $O(\alpha)$ of the mass. By our Kolmogorov distance result ([32, Lemma 4.15]), we have that $|F_\gamma(\mathcal{U}^{[m]\setminus S})|$ and the binomial distribution $\mathsf{Bin}(n, \gamma)$ assign similar mass to each of these intervals. Moreover, our continuity result ([32, Proposition 4.21]) implies the mass in a fixed interval is almost uniformly distributed. Thus for most weights w, $|F_\gamma(\mathcal{U}^{[m]\setminus S})|$ assigns roughly the same mass to w as $\mathsf{Bin}(n, \gamma)$ does. Summing over all weights provides an upper bound on the total variation distance between the two weight distributions. We remark that there is a slight complication in the case of $\gamma = 1/2$, as there [32, Proposition 4.21] only lets us compare weights that are an even distance apart. However, by further subdividing each interval we consider into its even and odd parts, we are able to carry out a similar argument as before.

Finally, we recall that $f(\mathcal{U}^m)$ is a mixture of the distributions $F_\gamma(\mathcal{U}^{[m]\setminus S})$. Since the weight distribution of each $F_\gamma(\mathcal{U}^{[m]\setminus S})$ is close to the weight distribution corresponding to one from Theorem 2, their mixture $|f(\mathcal{U}^m)|$ naturally is close to a mixture of those weight distributions (see [32, Lemma 4.24]). We conclude by recalling that Equation (1) implies it was sufficient to classify the output weight distribution.

2.1.5 Mixing Weights

The above argument shows that $f(\mathcal{U}^m)$ is close to a mixture $\mathcal{M}$ of the form specified in Theorem 2, but without the additional control on the mixing weights. The reason for this shortcoming is that the threshold A at which we consider an input bit "high degree" depends not just on d, but also on ε, and so the mixing weights in the obtained mixture also depend on ε. If we instead chose a threshold that only depended on d, we would not have been able to obtain an effective error bound in the Kolmogorov distance step (i.e., [32, Proposition 4.16]) when $\gamma = 1/2$.

The key observation is that the arguments of Step 1 only require conditioning on input bits of degree larger than $n/O_d(1)$. In other words, if we perform some different conditioning κ on the set of input bits $S' \subseteq [m]$ above the weaker threshold $n/O_d(1)$, the distributions generated by the restricted functions will still have output weights which strongly concentrate around some integer multiple of $n/2^d$. Moreover, the mixing weights in this setting *are* integer multiples of $2^{-O_d(1)}$.

It remains to argue that the mixing weights on $\mathcal{M}$, the mixture we proved $f(\mathcal{U}^m)$ is close to by the stronger conditioning ρ, correspond to the mixing weights derived from the weaker conditioning κ (see [32, Lemma 4.26]). Since both conditionings produce mixtures of the same distribution, we have

$$\sum_\gamma \Pr_\rho[\gamma_\rho = \gamma] \cdot \mathop{\mathbb{E}}_{\rho:\gamma_\rho=\gamma} f_\rho(\mathcal{U}^{[m]\setminus S}) = f(\mathcal{U}^m) = \sum_\gamma \Pr_\kappa[\gamma_\kappa = \gamma] \cdot \mathop{\mathbb{E}}_{\kappa:\gamma_\kappa=\gamma} f_\kappa(\mathcal{U}^{[m]\setminus S'}).$$

By standard concentration bounds, the probability that $x \sim f(\mathcal{U}^m)$ has Hamming weight close to γn is almost entirely determined by the mass assigned to the distributions whose output weight concentrates around γ. In other words, we know $\Pr_\rho[\gamma_\rho = \gamma] \approx \Pr_\kappa[\gamma_\kappa = \gamma]$

for each γ. By our earlier analysis and the assumption that $f(\mathcal{U}^m)$ is close to $\mathcal{D}$, it must be that the mixing weight on $\mathsf{Bin}(n,\gamma)$ in $\mathcal{M}$ is roughly $\Pr_\rho[\gamma_\rho = \gamma]$, so it must also be close to $\Pr_\kappa[\gamma_\kappa = \gamma]$, which has the form we sought.

The analysis in the case of $\gamma = 1/2$ is similar, but as always, we need to keep track of the output weight's parity. Here, the analogous equivalence is

$$\Pr_\rho[\gamma_\rho = 1/2 \wedge |f(\mathcal{U}^m)| \text{ is even}] \approx \frac{1}{2^{|S'|}} \sum_{\kappa:\gamma_\kappa = 1/2} \Pr_{x \sim \mathcal{U}^{[m]\setminus S'}}[|f_\kappa(x)| \text{ is even}]. \tag{2}$$

Since f is d-local with n output bits, the parity of $|f_\kappa(x)|$ can be represented as a degree-d $\mathbb{F}_2$-polynomial with $O_d(n)$ monomials. Thus, the mixing weight for the uniform distribution over n-bit strings of even Hamming weight in $\mathcal{M}$ must be close to the right-hand side of Equation (2), which is the precise sense in which the mixing weights are "determined by low-degree $\mathbb{F}_2$-polynomials". The odd parity case is essentially identical. This concludes the proof overview of Theorem 2.

2.2 Technical Comparison to Prior Works

We now briefly survey the approaches of related prior works. To avoid an overly verbose digression, we attempt to focus only on the most relevant papers and do not hope to be exhaustive. Similarly, we restrict our attention within the mentioned works to the results and techniques most pertinent to our own; most, if not all, of the papers discussed contain a number of other interesting results.

The study of the complexity of sampling goes back to at least the 1980s in the influential work of Jerrum, Valiant, and Vazirani [29]. In the context of shallow circuits, several clever sampling constructions (including the one in the introduction) were devised in [5, 10, 28], while the first serious treatment of the complexity of sampling (with shallow circuits) appeared in [40]. There, Viola proved an assortment of sampling-related results, but we will confine our attention to two on the hardness of approximately generating $\mathcal{D}_{n/2}$, the uniform distribution over n-bit strings of Hamming weight $n/2$. (This specific choice is purely for simplicity, and both results also apply to the setting of $\mathcal{D}_{\alpha n}$ for $\alpha \in (0,1)$.)

The first result we will mention [40, Theorem 1.6] is an unconditional lower bound of $2^{-O(d)} - O(1/n)$ on the distance between $f(\mathcal{U}^m)$, where $f\colon \{0,1\}^m \to \{0,1\}^n$ is a d-local[4] function, and $\mathcal{D}_{n/2}$. The proof, much like our own (see Step 2), relies on a Kolmogorov distance bound obtained from a k-wise independence assumption [19, 23]. If the output distribution $f(\mathcal{U}^m)$ is k-wise independent for some large integer k, then [23, Theorem I.5] implies it has the wrong Hamming weight with constant probability, and we are done. Otherwise, there exists a k-tuple $T \subseteq [n]$ of output bits that are not uniformly distributed over $\{0,1\}^k$. In this case, one observes that the probability assigned to any element of $\{0,1\}^k$ is an integer multiple of 2^{-kd}, so the marginal distribution over T must be at least 2^{-kd}-far from uniform. The proof concludes by noting that the marginal distribution of $\mathcal{D}_{n/2}$ onto T is very close to uniform. Note that we use similar granularity ideas in the proof of [32, Proposition 4.5] (in Step 1).

The second result [40, Theorem 1.3] provides a much stronger lower bound of $1 - 1/\mathsf{poly}(n)$ for the distance between $f(\mathcal{U}^m)$ and $\mathcal{D}_{n/2}$ for any $O(\log n)$-local function f, but is *conditional* on the input size m not substantially exceeding the information-theoretic minimum required to generate the target distribution. The proof begins by partitioning the input bits u of f as $u = (x, y)$ and expressing

[4] In fact, the result is proven for an adaptive version of locality.

$$f(u) = f(x, y) = h(y) \circ g_1(x_1, y) \circ \cdots \circ g_s(x_s, y),$$

where each g_i may depend arbitrarily on y but only on the single bit x_i of x. Additionally, each $g_i(x_i, y) \in \{0,1\}^{O(d)}$. By a greedy approach, this can be accomplished with $s \geq \Omega(n/d^2) = n/\mathsf{polylog}(n)$. The argument proceeds by conditioning on certain input bits (in this case y) and splitting into two scenarios depending on the amount of concentration; this strategy has been employed by several later works, including this one.

In the "concentrated" case where at least $\sqrt{n}$ many g_i's become fixed, the output distribution of $f(x, y)$ has small support. Even taking the union over all choices of such y, one finds that many elements in the support of $\mathcal{D}_{n/2}$ cannot be obtained. Note that this is where the input length restriction originates. Alternatively, at least $s - \sqrt{n} = n/\mathsf{polylog}(n)$ of the g_i's take multiple values. One would like to apply standard anticoncentration results to argue the output weight is too spread out, but as noted earlier in the proof overview (see Step 3), it may be that the Hamming weight of a g_i is fixed, even if the output is not. Viola circumvents this issue by adding an additional "test" to determine whether at most $2\sqrt{n}$ many g_i's can output the all zeros string; in this case, taking two or more values implies the weight cannot be fixed, and thus anticoncentration inequalities can be applied.

We now turn to the recent work of Filmus, Leigh, Riazanov, and Sokolov [21], who addressed the case of $\mathcal{D}_k$ for $k = o(n)$. They proved [21, Theorem 1.2] that $\widetilde{\Omega}(\log(n/k))$ bits of locality (even adaptively chosen) are required to generate $\mathcal{D}_k$ to constant error. Moreover, the same result holds for the uniform distribution over strings whose Hamming weight lies in a set S, where $\max_{s \in S} s = k$. Their proof first reduces to the case of considering $\mathcal{D}_1$, and then proceeds via a hitting set vs independent set dichotomy. This can again be viewed as fitting into the concentration vs anticoncentration paradigm. If there are $O(d2^d)$ input bits that together affect every output bit, then one can afford to condition on them and induct on the remaining $(d-1)$-local sub-distributions. Bounds from these distributions can eventually be recombined to deduce bounds on the full distribution using some version of a union bound, such as [32, Lemma 3.4]. Otherwise, there are $\Omega(2^d)$ independent output bits, and one can argue it is very likely that two of these bits evaluate to 1, since each (nonzero) output bit is 1 with probability at least 2^{-d}. Interestingly, the authors are able to use the *robust sunflower lemma* [36, 4] from extremal combinatorics to improve their quantitative bounds; we refer the reader to their original paper for details.

Also recently, strong unconditional locality lower bounds for sampling $\mathcal{D}_{\alpha n}$ where α is a non-dyadic rational were proven independently in [44, 30]. At a very high level, both proofs follow from the observation (recorded in [39]) that the marginal distribution on any output bit of a d-local function f must have probabilities that are integer multiples of 2^{-d}, which cannot accurately approximate (say) $1/3$, corresponding to the marginal distribution for $\mathcal{D}_{n/3}$. We begin by describing the approach of Viola [44, Theorem 25], continuing to focus on the case of $\alpha = 1/3$ for simplicity. First assume by contradiction $\left\| f(\mathcal{U}^m) - \mathcal{D}_{n/3} \right\|_{\mathsf{TV}} \leq 1 - \varepsilon$ for some ε to be determined. The proof proceeds by arguing that there must exist a large subset $R \subseteq \{0,1\}^m$ of the inputs, such that $f(\mathcal{U}(R))$, f evaluated on inputs drawn uniformly at random from R, lands entirely in the support of $\mathcal{D}_{n/3}$ and has almost full min-entropy.[5] The so-called *fixed-set lemma* of Grinberg, Shaltiel, and Viola [25] then implies that R can be further restricted to a large subset $R' \subseteq R$ whose uniform distribution is nearly indistinguishable by d-local functions from a product distribution where each bit is either

[5] Recall the *min-entropy* of a random variable X is $\log(1/\max_{x \in \mathsf{supp}(X)} \Pr[X = x])$.

fixed or uniform. Then the large min-entropy of $f(\mathcal{U}(R))$ (and thus $f(\mathcal{U}(R'))$) combined with the fact that the distribution is contained in the support of $\mathcal{D}_{n/3}$ implies there must be at least one output bit whose marginal distribution is extremely close to 1/3. This, however, contradicts the fact that 1/3 cannot accurately be approximated by an integer multiple of 2^{-d}.

The proof in [30] instead operates very similarly to the details of Step 3. Through a graph-theoretic lemma [30, Corollary 4.8], they show that there must be a set of at most $r/2^d$ many input bits upon whose conditioning results in r many independent output bits for $r \approx n/2^{d^2}$. As previously noted, each of these output bits inevitably incurs some error from mismatched marginal distributions, and these errors can be aggregated via standard concentration inequalities (as in [32, Lemma 3.3]). Bounds from the individual sub-distributions can once again be recombined via a union bound result like [32, Lemma 3.4].

The work [30] also contains locality bounds for $\mathcal{D}_{\alpha n}$ when α is a dyadic rational. The proof has strong similarities to the non-dyadic case, except that now the marginal distributions on output bits of f do not necessarily disagree with those of the target distribution. The authors overcome this issue by turning to the familiar concentration vs. anticoncentration paradigm. The details have almost entirely been spelled out already in Step 3, as our analysis is nearly identical at that part. We briefly note that the quantitative dependencies here are much worse than in the non-dyadic case, essentially for the reason that one needs to obtain a much richer structure than simply independent output bits, and so the analogous graph-theoretic lemma in this case has poor bounds.[6]

Finally, let us mention the predecessor of this work [31], which classifies what uniform symmetric distributions can be sampled by functions of bounded locality, confirming a conjecture of [21]. In broad strokes, it follows similarly to the proof overview, so we will be brief. Through a strategic conditioning of input bits, one may reduce to the case where no input bits have large degree and almost all the output bits are k-wise independent. Then one can show the resulting Hamming weight distribution is close in Kolmogorov distance to the binomial distribution, and it satisfies a continuity-type property. Combining these steps together, one classifies the Hamming weight distribution, which is enough to classify the actual distribution, since we assume $f(\mathcal{U}^m)$ is close to a uniform symmetric distribution. Much of the present work is an extension of the ideas in [31] to the general symmetric case, although there are several parts (e.g., much of the analysis in Step 1 and the finer control on the mixing weights) that require substantially new ideas.

We conclude by noting that several works [35, 6, 42, 43, 11] prove hardness results against the more powerful classes of AC^0 circuits or read-once branching programs. There is a reasonable overlap in the techniques used with those discussed above, but the proofs of all these results rely on the special properties of certain pseudorandom objects like good codes [35, 6, 11] or extractors [41, 43].

References

1 M. Ajtai. Σ^1_1-formulae on finite structures. *Ann. Pure Appl. Logic*, 24(1):1–48, 1983. doi:10.1016/0168-0072(83)90038-6.

2 Yaroslav Alekseev, Mika Göös, Konstantin Myasnikov, Artur Riazanov, and Dmitry Sokolov. Sampling permutations with cell probes is hard. In *Proceedings of the 58th Annual ACM Symposium on Theory of Computing*, 2026.

[6] Moreover, the bounds in this graph-theoretic lemma are essentially best possible; see [30, Appendix A].

3 Maryam Aliakbarpour, Eric Blais, and Ronitt Rubinfeld. Learning and testing junta distributions. In *Conference on Learning Theory*, pages 19–46. PMLR, 2016. URL: http://proceedings.mlr.press/v49/aliakbarpour16.html.

4 Ryan Alweiss, Shachar Lovett, Kewen Wu, and Jiapeng Zhang. Improved bounds for the sunflower lemma. *Annals of Mathematics*, 194(3):795–815, 2021.

5 László Babai. Random oracles separate PSPACE from the polynomial-time hierarchy. *Information Processing Letters*, 26(1):51–53, 1987. doi:10.1016/0020-0190(87)90036-6.

6 Chris Beck, Russell Impagliazzo, and Shachar Lovett. Large deviation bounds for decision trees and sampling lower bounds for AC0-circuits. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 101–110. IEEE, 2012. doi:10.1109/FOCS.2012.82.

7 Adam Bene Watts and Natalie Parham. Unconditional Quantum Advantage for Sampling with Shallow Circuits. In Shubhangi Saraf, editor, *17th Innovations in Theoretical Computer Science Conference (ITCS 2026)*, volume 362 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 17:1–17:12, Dagstuhl, Germany, 2026. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2026.17.

8 Lucien Birgé. Estimating a density under order restrictions: Nonasymptotic minimax risk. *The Annals of Statistics*, pages 995–1012, 1987.

9 Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989. doi:10.1145/76359.76371.

10 Ravi B Boppana and Jeffrey C Lagarias. One-way functions and circuit complexity. *Information and Computation*, 74(3):226–240, 1987. doi:10.1016/0890-5401(87)90022-8.

11 Eshan Chattopadhyay, Jesse Goodman, and David Zuckerman. The space complexity of sampling. In *13th Innovations in Theoretical Computer Science Conference,(ITCS 2022)*, 2022.

12 Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, Shachar Lovett, and David Zuckerman. XOR lemmas for resilient functions against polynomials. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 234–246, 2020. doi:10.1145/3357713.3384242.

13 Eshan Chattopadhyay and David Zuckerman. Explicit two-source extractors and resilient functions. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 670–683, 2016. doi:10.1145/2897518.2897528.

14 Gil Cohen and Leonard J Schulman. Extractors for near logarithmic min-entropy. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 178–187. IEEE, 2016. doi:10.1109/FOCS.2016.27.

15 Constantinos Daskalakis, Ilias Diakonikolas, Ryan O'Donnell, Rocco A Servedio, and Li-Yang Tan. Learning sums of independent integer random variables. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 217–226. IEEE, 2013. doi:10.1109/FOCS.2013.31.

16 Constantinos Daskalakis and Gautam Kamath. Faster and sample near-optimal algorithms for proper learning mixtures of Gaussians. In *Conference on Learning Theory*, pages 1183–1213. PMLR, 2014. URL: http://proceedings.mlr.press/v35/daskalakis14.html.

17 Anindya De and Thomas Watson. Extractors and lower bounds for locally samplable sources. *ACM Transactions on Computation Theory (TOCT)*, 4(1):1–21, 2012. doi:10.1145/2141938.2141941.

18 Ilias Diakonikolas. Learning structured distributions. *Handbook of Big Data*, 267:10–1201, 2016.

19 Ilias Diakonikolas, Parikshit Gopalan, Ragesh Jaiswal, Rocco A Servedio, and Emanuele Viola. Bounded independence fools halfspaces. *SIAM Journal on Computing*, 39(8):3441–3462, 2010. doi:10.1137/100783030.

20 Lutz Dümbgen and Kaspar Rufibach. Maximum likelihood estimation of a log-concave density and its distribution function: Basic properties and uniform consistency. *Bernoulli*, pages 40–68, 2009.

21 Yuval Filmus, Itai Leigh, Artur Riazanov, and Dmitry Sokolov. Sampling and certifying symmetric functions. In *Approximation, Randomization, and Combinatorial Optimization. (APPROX/RANDOM)*, 2023.

22 Merrick Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Math. Systems Theory*, 17(1):13–27, 1984. doi:10.1007/BF01744431.

23 Parikshit Gopalan, Ryan O'Donnell, Yi Wu, and David Zuckerman. Fooling functions of halfspaces under product distributions. In *2010 IEEE 25th Annual Conference on Computational Complexity*, pages 223–234. IEEE, 2010. doi:10.1109/CCC.2010.29.

24 Daniel Grier, Daniel M. Kane, Jackson Morris, Anthony Ostuni, and Kewen Wu. Quantum Advantage from Sampling Shallow Circuits: Beyond Hardness of Marginals. In Shubhangi Saraf, editor, *17th Innovations in Theoretical Computer Science Conference (ITCS 2026)*, volume 362 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 73:1–73:14, Dagstuhl, Germany, 2026. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2026.73.

25 Aryeh Grinberg, Ronen Shaltiel, and Emanuele Viola. Indistinguishability by adaptive procedures with advice, and lower bounds on hardness amplification proofs. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 956–966. IEEE, 2018. doi:10.1109/FOCS.2018.00094.

26 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 6–20, 1986. doi:10.1145/12130.12132.

27 Johan Håstad. *Computational limitations for small depth circuits*. PhD thesis, Massachusetts Institute of Technology, 1986.

28 Russell Impagliazzo and Moni Naor. Efficient cryptographic schemes provably as secure as subset sum. *Journal of cryptology*, 9(4):199–216, 1996. doi:10.1007/BF00189260.

29 Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical computer science*, 43:169–188, 1986. doi:10.1016/0304-3975(86)90174-X.

30 Daniel M Kane, Anthony Ostuni, and Kewen Wu. Locality bounds for sampling Hamming slices. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1279–1286, 2024. doi:10.1145/3618260.3649670.

31 Daniel M Kane, Anthony Ostuni, and Kewen Wu. Locally sampleable uniform symmetric distributions. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 1807–1816, 2025. doi:10.1145/3717823.3718243.

32 Daniel M Kane, Anthony Ostuni, and Kewen Wu. Symmetric distributions from shallow circuits. *arXiv preprint arXiv:2511.14127*, 2025. doi:10.48550/arXiv.2511.14127.

33 Michael Kearns, Yishay Mansour, Dana Ron, Ronitt Rubinfeld, Robert E Schapire, and Linda Sellie. On the learnability of discrete distributions. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 273–282, 1994. doi:10.1145/195058.195155.

34 Bruce G Lindsay. Mixture models: Theory, geometry and applications. In *NSF-CBMS Regional Conference Series in Probability and Statistics*, pages i–163. JSTOR, 1995.

35 Shachar Lovett and Emanuele Viola. Bounded-depth circuits cannot sample good codes. In *2011 IEEE 26th Annual Conference on Computational Complexity*, pages 243–251. IEEE, 2011. doi:10.1109/CCC.2011.11.

36 Benjamin Rossman. The monotone complexity of k-clique on random graphs. *SIAM Journal on Computing*, 43(1):256–279, 2014. doi:10.1137/110839059.

37 Ronen Shaltiel and Jad Silbak. Explicit codes for poly-size circuits and functions that are hard to sample on low entropy distributions. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 2028–2038, 2024. doi:10.1145/3618260.3649735.

38 Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984. doi:10.1145/1968.1972.

39 Emanuele Viola. Bit-probe lower bounds for succinct data structures. *SIAM Journal on Computing*, 41(6):1593, 2012.

40 Emanuele Viola. The complexity of distributions. *SIAM Journal on Computing*, 41(1):191–218, 2012. doi:10.1137/100814998.

41 Emanuele Viola. Extractors for Turing-machine sources. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 663–671. Springer, 2012. doi:10.1007/978-3-642-32512-0_56.

42 Emanuele Viola. Extractors for circuit sources. *SIAM Journal on Computing*, 43(2):655–672, 2014. doi:10.1137/11085983X.

43 Emanuele Viola. Sampling lower bounds: boolean average-case and permutations. *SIAM Journal on Computing*, 49(1):119–137, 2020. doi:10.1137/18M1198405.

44 Emanuele Viola. New sampling lower bounds via the separator. In *38th Computational Complexity Conference (CCC 2023)*, 2023. doi:10.4230/LIPIcs.CCC.2023.26.

45 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles. In *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, pages 1–10. IEEE, 1985.

46 Yannis G Yatracos. Rates of convergence of minimum distance estimators and kolmogorov's entropy. *The Annals of Statistics*, 13(2):768–774, 1985.

47 Huacheng Yu and Wei Zhan. Sampling, Flowers and Communication. In *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*, volume 287 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 100:1–100:11. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ITCS.2024.100.

Load Balancing Under Adaptive Bin Deletions

Haim Kaplan
Tel Aviv University, Israel
Google Research, Tel Aviv, Israel

Shay Sapir
Weizmann Institute, Rehovot, Israel
Google Research, Tel Aviv, Israel

Uri Stemmer
Tel Aviv University, Israel
Google Research, Tel Aviv, Israel

Abstract

We analyze a balls-and-bins game against an adaptive adversary that sequentially deletes bins. Starting with n balls distributed across n bins, the adversary deletes a bin in each step, forcing the algorithm to redistribute its balls to surviving bins. We prove that after $n/2$ rounds, uniform random redistribution yields optimal $O(n)$ recourse and $O(\frac{\log n}{\log \log n})$ maximum load. Furthermore, we show that applying the "power of two choices" reduces the maximum load to $O(\log \log n)$ while maintaining linear recourse.

We also consider a variation of this game where the balls from the deleted bin are partitioned evenly among $d \ll n$ random bins rather than being redistributed independently. We demonstrate that keeping the balls together ($d = 1$), which gives small maximum load and recourse against an oblivious adversary, fails against an adaptive adversary. Nevertheless, we show that splitting the balls into just two groups ($d = 2$) is sufficient to recover linear recourse and efficient load balancing in the adaptive setting.

2012 ACM Subject Classification Theory of computation → Adversary models; Theory of computation → Randomness, geometry and discrete structures

Keywords and phrases Balls and Bins, Adaptive Adversary

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.54

Category RANDOM

Funding *Haim Kaplan*: Partially supported by the Israel Science Foundation (grant 1156/23), and the Blavatnik Research Foundation.
Uri Stemmer: Partially supported by the Israel Science Foundation (grant 1419/24), and the Blavatnik Research Foundation.

1 Introduction

Balls-and-bins processes serve as a fundamental abstraction for studying load balancing problems, with diverse applications across numerous domains [2, 22, 27, 25, 32, 17, 31]. In the simplest version of the problem, n balls are assigned to n bins. Here it is well established that distributing every ball independently and uniformly over the bins yields a maximum load of $O(\frac{\log n}{\log \log n})$ [19]. Furthermore, utilizing the "power of two choices", where balls are placed sequentially in the least loaded of two randomly chosen bins, dramatically reduces the maximum load to $O(\log \log n)$ [3].

Beyond these classical settings, the landscape of balls-and-bins problems is vast. Variations include settings with weighted balls [13, 26], heterogeneous bin capacities [10, 32], and graph-based constraints where balls can probe only specific subsets of bins [23, 18]. Berenbrink et al. [11, 12] extended the power of two choices from the *lightly-loaded* case studied by [3],

© Haim Kaplan, Shay Sapir, and Uri Stemmer;
licensed under Creative Commons License CC-BY 4.0

where the number of balls m is equal to the number of bins n, to the *heavily-loaded* case, where $m \gg n$. Remarkably, they showed that the additive gap between the maximum load and the average load remains the same as in the lightly-loaded case. Los and Sauerwald [24] analyzed the power of two choices in an "incomplete information" setting, investigating scenarios where algorithms must rely on load estimates rather than exact values. Vöcking [30] considered multiple-choice settings (rather than just two choices), where balls are placed in the least loaded of d selected bins, demonstrating that a nonuniform selection of the d locations leads to better load balancing.

Of particular relevance to our work are *dynamic* variations of balls-and-bins, where the goal is to maintain a balanced allocation over time as balls or bins arrive and depart. This line of work was initiated by [3], who considered a model where balls are added and deleted *at random*. They showed that their "power of two choices" extends to this dynamic setting, maintaining a maximum load of $O(\log \log n)$ in the lightly-loaded case. Subsequent works [15, 30, 4] generalized this to the heavily-loaded case. Furthermore, these works allowed the sequence of operations to be determined by an *oblivious adversary*, that is, an adversary that fixes the full sequence of ball insertions and deletions before the process begins. Note that the random case studied by [3] is a special instance of an oblivious adversary.

Adaptive adversary

Unlike an oblivious adversary, who commits to the sequence of operations before the game begins, an adaptive adversary chooses its actions sequentially, potentially basing its decisions on the entire history of the execution. Designing algorithms and data structures that cope with adaptive adversaries has been the focus of much recent work across various sub-fields of theoretical computer science, including streaming algorithms, dynamic graph algorithms, and learning theory [8, 7, 21, 29, 6, 14, 1, 20]. However, in the context of dynamic balls-and-bins, to our knowledge, only two works have addressed general adaptive adversaries: Bender et al. [9] and Fine et al. [16].

Another work by Becchetti et al. [5], considered a repeated balls-and-bins game, which can be viewed as *specific adaptive strategy*. In this game, n balls are arbitrarily assigned to n bins, and in each round, one ball is extracted from each non-empty bin, and re-assigned to a uniformly random bin. Becchetti et al. [5] proved that this game admits a self-stabilizing property: after $O(n)$ rounds, the maximum load is $O(\log n)$, and remains $O(\log n)$ for $\text{poly}(n)$ rounds, with high probability.[1] However, the focus of Becchetti et al. [5] is on that specific adaptive strategy, hence it does not bound general adaptive adversaries.

Bender et al. [9] considered an adaptive *ball-recycling* game where m balls are initially distributed across n bins. At each time step, an adversary adaptively selects a bin, removes all its balls, and re-throws them at random. The goal of the adversary is to maximize the expected number of balls re-thrown per step in the stationary distribution (when it exists), a metric we refer to as the *long-run average recourse*. Bender et al. [9] established nearly tight bounds for this setting: there exists an adversary that achieves a long-run average recourse of $2m/(n+1)$, while no adversary can achieve long-run average recourse greater than $2m/n + 1$. However, since the work of [9] focuses on the stationary distribution, it does not yield strong bounds on the achievable recourse in finite-horizon games.

This gap was addressed by Fine et al. [16], who considered a variant of this game in which bins are *deleted* rather than *recycled*. This implies that the game can last at most n steps. This is formulated as a two-player game between an (adaptive) adversary and a randomized algorithm, defined as follows.

[1] High probability means with probability at least $1/\text{poly}(n)$.

▶ **Definition 1.1** (The Bin-Deletion Game). *Initially, the algorithm distributes n balls into n bins according to its initialization strategy (in our setting, this will either be the uniform distribution or using the power of two choices). Then the game proceeds for $T \leq n$ rounds, where in each round:*

1. *The adversary inspects the current assignment and selects a bin to delete.*
2. *The algorithm redistributes the balls from the deleted bin among the surviving bins.*

The algorithm has two primary objectives: to minimize the total number of balls it redistributes, referred to as the (total) recourse*, and to minimize the* maximum load *of any single bin throughout the game.*

▶ Remark 1.2. As is standard in the literature on balls-and-bins, we focus on "lightweight" strategies for the algorithm, where the destination of a ball is chosen either independently of the current system state or by probing only a constant number of randomly chosen bins. Such strategies are crucial in distributed settings where querying the load of every bin may be slow or expensive.

Fine et al. [16] considered this game in the case where $T = n - 1$ (i.e., the game continues until only one bin remains). They showed that no adversary can force more than $O(n \log n)$ total recourse against the simple algorithm that redistributes balls from deleted bins uniformly at random. This is optimal for $T = n - 1$, because at any step with r remaining bins, the adversary can force a recourse of at least n/r (the average load). Thus, even over just the last $n/2$ steps, any algorithm must incur a total recourse of:

$$\sum_{r=1}^{n/2} \frac{n}{r} = n \sum_{r=1}^{n/2} \frac{1}{r} = \Theta(n \log n).$$

Note that this argument fails when the number of deletions T is smaller than $n/2$. The analysis of recourse and maximum load when the adversary removes only a subset of the bins was left open by Fine et al. [16]. (The maximum load is irrelevant when $T = n - 1$, as it is trivially n when only one bin remains.)

1.1 Our results

We analyze the recourse and the maximum load of several algorithmic strategies for the adaptive bin deletion game (Definition 1.1). First, we resolve the behavior of this game for partial deletions. For simplicity of presentation, we only discuss here $T = \frac{n}{2}$ rounds, and present the extension to $T > \frac{n}{2}$ in Section 3. We establish that after $\frac{n}{2}$ rounds, independent uniform redistribution yields optimal $O(n)$ recourse and a maximum load of $O(\frac{\log n}{\log \log n})$, matching the classical static bounds. Moreover, we show that the power of two choices carries over to this adaptive setting, reducing the maximum load to $O(\log \log n)$ and recovering the exponential improvement seen in the static case. Specifically,

▶ **Theorem 1.3** (Informal). *In the adaptive bin-deletion game, after $T = \frac{n}{2}$ rounds, with high probability,*

1. *If balls are distributed independently and uniformly at random, the recourse is $O(n)$ and the maximum load is $O(\frac{\log n}{\log \log n})$, and*
2. *If balls are distributed using the two choices allocation scheme, the recourse is $O(n)$ and the maximum load is $O(\log \log n)$.*

Theorem 1.3 assumes that a deleted bin can send every one of its balls independently into a different random (remaining) bin. However, this might sometimes be infeasible. For example, in a communication network, this might require a server to connect to a prohibitively

large number of machines. Therefore, we seek to reduce the number of target bins that a deleted bin connects to. We consider a restrictive algorithmic strategy called *d-split*: when a bin is deleted, we partition its balls among d randomly chosen bins. This strategy was suggested in [16], who asked what is the recourse of this scheme, particularly when $d = 2$. In Section 4, we answer this question when the game is played for $T = n/2$ rounds, showing that this d-split strategy suffices for guaranteeing linear recourse and nearly-optimal maximum load, even for $d = 2$. The analysis of this variant is significantly more challenging, and it may provide insights for handling adaptive inputs in other settings. Extending our result to $T = (1 - o(1))n$ rounds remains open. Note that for this variant, our bound on the maximum load holds only with constant probability.

▶ **Theorem 1.4.** *In 2-split against an adaptive adversary, after $\frac{n}{2}$ rounds, the recourse is $O(n)$ with high probability, and the maximum load is* $\mathrm{poly}(\log n)$*, with constant probability.*

As noted by [16], restricting this variant further to $d = 1$, which forces a deleted bin to send all its balls into *one* randomly chosen bin, is disastrous against an adaptive adversary: The adversary can always delete the heaviest bin, causing a snowball effect where a heavy packet of balls repeatedly merges with other bins. Already after $n/2$ rounds, this results in maximum load $\Omega(n)$ and recourse $\Omega(n^2)$.

Somewhat surprisingly, we show that this restrictive variant (with $d = 1$) is actually a solid strategy against an *oblivious* adversary that chooses the sequence of deletions in advance. This underscores the power of the adaptive adversary and sets it apart from the oblivious adversary in the context of dynamic balls-and-bins. Specifically, in Section 5 we prove the following theorem:

▶ **Theorem 1.5.** *In 1-split against an oblivious adversary, after $\frac{n}{2}$ rounds, the recourse is $O(n \log n)$ and the maximum load is $O(\log n)$, with high probability.*

To grasp the intuition behind this result, consider a random *forest* graph on n vertices constructed as follows: for each step i from 1 to $n/2$, vertex i uniformly samples a parent $j \in \{i + 1, \ldots, n\}$ and becomes a child of j. This process yields a forest with $n/2$ roots (the vertices indexed $n/2 + 1$ through n).

Critically, the recourse in our oblivious $d = 1$ game is exactly equal to the sum of the depths of all vertices in this forest, and the maximum load is the size of the largest tree. This equivalence holds because, without loss of generality, an oblivious adversary deletes bins 1 through $n/2$ sequentially. When bin i is deleted, it merges into a surviving bin $j > i$, creating the directed edge (i, j). Consequently, a ball starting at bin u incurs a movement cost exactly equal to the length of the path from u to a root; and the load of bin i is the size of the tree rooted at i. We provide a direct analysis showing that, for such a random forest, the total sum of depths is $O(n \log n)$, and the maximum tree size is $O(\log n)$ with high probability.

1.2 Technical overview

Our proofs use coupling arguments to show stochastic domination, formalized as follows.

▶ **Definition 1.6** (First-order stochastic domination)**.** *Random variable A has first-order stochastic dominance over random variable B, denoted $A \succeq B$ if, for all $x \in \mathbb{R}$,*

$$\Pr(A \geq x) \geq \Pr(B \geq x).$$

▶ **Lemma 1.7** (Theorem 4.2.3 of [28].). *For random variables A, B, we have $A \succeq B$ if and only if there exist random variables A', B', such that A' has the same distribution as A, and B' has the same distribution as B, and $A' \geq B'$. The random variables A', B' are called a monotone coupling.*

For brevity, from now on we omit the term "first-order" when referring to Definition 1.6, and omit the word "monotone" when referring to Lemma 1.7. For concreteness, suppose that the number of rounds is $n/2$, for all variants.

Uniform game

Consider the uniform algorithm, that for each ball, draws a bin uniformly and independently at random from the surviving bins, and assigns the ball to the drawn bin. We show that the recourse and maximum load in this adaptive game are stochastically dominated by the recourse and maximum load in a related *non-adaptive game*, and then bound the recourse and maximum load of the non-adaptive game using standard tools. The non-adaptive game is the following: throw balls independently and uniformly at random to n bins, until every subset of $n/2$ bins contains at least n balls. It is not difficult to show that with high probability, the number of balls in this game is at most $10n$, and the bound on the maximum load follows by classic results, for a standard balls and bins problem with $m = 10n$.

The coupling argument, which is proven in Section 3, is as follows. Whenever we want to throw a ball in the adaptive game, we do the following. In the non-adaptive game, throw a ball uniformly at random over the n bins. If the ball lands in a bin of the same index as a surviving bin in the adaptive game, place the ball in the adaptive game at that corresponding bin. Otherwise, repeat this and throw another ball in the non-adaptive game. We obtain that the configuration, at the bins which correspond to surviving bins in the adaptive game, is the same in both games. Moreover, the non-adaptive game may only end after round $n/2$ of the adaptive game, since at round $n/2 - 1$, there are $n/2 + 1$ surviving bins, which contain n balls, and so there is a set of $n/2$ bins with strictly less than n balls. Thus, the configuration in the non-adaptive game is point-wise larger than the configuration in the adaptive game, and the proof follows.

▶ Remark 1.8. It is necessary that the algorithm's coins are hidden from the adversary, at least in order to bound the maximum load. To see this, we first define a model where the algorithm's random coins are set in advance and revealed to the adversary. Suppose that every bin b holds a sequence σ_b of independent random variables drawn from the uniform distribution over $[n]$. When bin b is deleted, it sends its balls to surviving bins in σ_b, in the order of σ_b (this is basically rejection sampling).

Given the sequences of all bins, the adversary can do the following attack. Look at the first $\sqrt{n}$ random variables of the first $\sqrt{n}$ bins. With constant probability, there exists $i \in [\sqrt{n}]$ such that n appears in the first $\sqrt{n}$ random variables of i. Delete the other $\sqrt{n}$ bins indexed by random variables in σ_i, and then delete i. This yields an additional $O(1)$ expected load on bin n. Repeat $O(\sqrt{n})$ times, resulting in expected load $\Omega(\sqrt{n})$ in bin n after $n/2$ rounds.

Two choices

In the adaptive two choices paradigm, each ball draws two bins independently and uniformly over the surviving bins, and picks the least loaded bin. Our proof for this setting is similar to that of the uniform game, except that the non-adaptive game we reduce to (via coupling) is different. Specifically, we consider the following non-adaptive variant of the two choices game

(without an adversary): Start with n empty bins. Then, iteratively pick *pairs of indices* (i, j) uniformly from $[n] \times [n]$, and place a ball in the least loaded among bins i, j. The game proceeds until for every subset $S \subseteq [n]$ of size $|S| = n/2$ we have that $S \times S \subset [n] \times [n]$ contains at least n pairs. We show that the recourse and maximum load in the *adaptive* two choices game are stochastically dominated by the number of rounds and the maximum load in the *non-adaptive* two choices game, and then bound these parameters in the non-adaptive game.

2-split, bounding the recourse

Recall that with the 2-split strategy, whenever a bin is deleted, its balls are split into two packets, and each packet is independently thrown to a uniformly random surviving bin. Theorem 1.4 states that after $\frac{n}{2}$ rounds, the recourse is $O(n)$ and the maximum load is $\text{poly}(\log n)$. We first discuss the recourse bound. To this end, we relate the adaptive 2-split game to an intermediate (adaptive) game, denoted π_{pay}, where: (1) balls from deleted bins are thrown independently uniformly at random, instead of using the 2-split strategy; and (2) when a bin with k balls is deleted, the incurred recourse is $k + O(k \log k)$ instead of k. Note that the first modification "helps us" in that it makes the problem easier to analyze and the second modification "helps the adversary" as the recourse is increased.

Our analysis proceeds using two main claims: (1) The recourse in π_{pay} is at total variation distance $\frac{1}{n^8}$ from a random variable that stochastically dominates the recourse in 2-split; and (2) the recourse in π_{pay} is $O(n)$ with high probability. By a union bound, the recourse in 2-split is $O(n)$ with high probability. The proof of the second claim is very similar to the uniform game, and is omitted from this overview. The basic idea behind the first claim is that if we consider $k \geq 2$ balls that move because of a single deletion, and track their movement until each of the k balls become isolated from the other $k-1$ balls, then with high probability, they do not inflict a lot of recourse (concretely, $O(k \log k)$ recourse). Therefore, we can basically replace each deletion by paying this "cost", and throw the balls independently. This is formulated using the following toy problem and an inductive argument. In the toy problem, there are $2 \leq k \leq O(\frac{\log n}{\log \log n})$ balls, initially located at the same bin. In each round, the adversary deletes a bin that contains at least 2 balls, and the balls are split evenly to two sets that are thrown independently. The game ends when the adversary cannot delete bins. In Section 4.1, we show that with high probability, the recourse in the toy problem is $O(k \log k)$, and complete the proof using an inductive argument.

2-split, maximum load

To show that after $\frac{n}{2}$ rounds the maximum load is $\text{poly}(\log n)$, we consider the following exponential potential function. For a load vector $(L_1, \ldots, L_n)$ and a set $I \subseteq [n]$ denoting the set of surviving bins, the potential is

$$w = \sum_{i \in I} e^{\alpha L_i},$$

where $\alpha = \frac{1}{\log n}$. By tracking w, we obtain that there exists a constant $C > 1$, such that with probability at least $1/2$, when the maximum load increases by a $\log n$ factor, at least n/C rounds passed. We repeatedly apply such an argument, and obtain that every time the maximum load increases by a $\log n$ factor, with probability $1/2$, at least n/C rounds passed. The $\text{poly}(\log n)$ bound on the maximum load after $n/2$ rounds then follows by standard arguments.

2 Preliminaries

Our proofs use the following concentration bounds.

▶ **Lemma 2.1** (Hoeffding's inequality). *Let $X_1, \ldots, X_n$ be independent random variables, such that $0 \le X_i \le C$ for all i. Then,*

$$\Pr\Big(\Big|\sum_{i=1}^{n} X_i - \sum_{i=1}^{n} \mathbb{E}X_i\Big| > t\Big) \le 2\exp\Big(-\frac{2t^2}{nC^2}\Big).$$

▶ **Lemma 2.2** (Bernstein's inequality). *Let $X_1, \ldots, X_n$ be independent random variables, such that $0 \le X_i \le C$ for all i. Denote $S = \sum_{i=1}^{n} X_i$ and $V = \mathrm{var}(S)$. Then,*

$$\Pr(|S - \mathbb{E}S| > t) \le 2\exp\Big(-\frac{t^2/2}{V + Ct/3}\Big).$$

3 Uniform and two choices allocations

Recall that we consider a 2-player game between an adaptive adversary and an algorithm. Initially, there are n balls independently distributed uniformly at random over n bins. The adversary sees the assignment of balls into bins, and picks a bin to delete. The algorithm must distribute the balls from the deleted bin among the remaining bins. This game continues in this manner for T rounds. The algorithm has two objectives: to minimize the number of balls that it distributes, called recourse, and to minimize the maximum load. We first consider the algorithm that independently distributes the balls from the deleted bin uniformly at random among the remaining bins (for brevity, we simply say the balls are thrown to the remaining bins), which we call the uniform game. Previously, [16] considered this game for $T = n - 1$, i.e., the game continues until only one bin remains. They showed that the recourse is $O(n \log n)$, which is the best possible for any algorithm. We consider this game for intermediate number of rounds, and show the following.

▶ **Theorem 3.1.** *Let $T \in [n-1]$, and denote $\ell = \lceil \log \frac{n}{n-T} \rceil$. In T rounds of the uniform game, with high probability, the recourse is $O(\ell \cdot n)$ and the maximum load is*

$$\begin{cases} O(\log n \log \frac{\log \log n}{\log \log n - \ell}) & \text{if } \ell < \log \log n \text{ (corresponding to the lightly-loaded case),} \\ O(2^\ell + \log n \log \log \log n) & \text{if } \ell \ge \log \log n, \text{ (corresponding to the heavily-loaded case).} \end{cases}$$

▶ Remark 3.2. Put differently, the maximum load is

$$\begin{cases} O(\frac{\ell \log n}{\log \log n}) & \text{if } \ell \le \frac{1}{2} \log \log n, \\ O(\log n \log \frac{\log \log n}{\log \log n - \ell}) & \text{if } \frac{1}{2} \log \log n < \ell < \log \log n, \\ O(\log n \log \log \log n) & \text{if } \log \log n \le \ell < \log \log n + \log \log \log \log n, \\ O(2^\ell) & \text{if } \ell \ge \log \log n + \log \log \log \log n. \end{cases}$$

That is, for small ℓ it interpolates from the standard bound $O(\frac{\log n}{\log \log n})$ of the lightly-loaded case to $O(\log n \log \log \log n)$, and for large $\ell \ge \log \log n + \log \log \log \log n$, the maximum load is the same order as the average load. Recall that when throwing $m' = n$ balls into $n' = n/2^\ell$ bins uniformly at random, by [27], the maximum load is

$$\begin{cases} O\Big(\frac{\log n}{\log \frac{\log n}{2^\ell}}\Big) & \text{if } \ell < \log \log n \text{ (light load),} \\ 2^\ell + O(\sqrt{2^\ell \log n}) & \text{if } \ell \ge \log \log n \text{ (heavy load),} \end{cases}$$

hence our bound for the maximum load is nearly tight, for all regimes.

The recourse bound is the best possible, for any algorithm, as explained below, and matches the bound of [16] for $T = n - 1$.

The proof of Theorem 3.1 considers ℓ phases. The j-th phase, where $j \in [\ell]$, is all rounds numbered in $[n - \frac{n}{2^{j-1}} + 1, n - \frac{n}{2^j}]$, i.e., the first phase is the first $n/2$ rounds, the second phase is the next $n/4$ rounds, etc. Phase j consists of exactly $n/2^j$ rounds and begins with $n/2^{j-1}$ alive bins. Consider the balls that are moved in the j-th phase, and denote by X_j their recourse, and by A_j the maximum load they incur. Clearly, the total recourse is $X = \sum_{j=1}^{\ell} X_j$, and the maximum load is $A \leq \sum_{j=1}^{\ell} A_j$. We show that with high probability, for all $j \in [\ell]$,

- $X_j = O(n)$, and
- $A_j = \begin{cases} O\left(\frac{\log n}{\log \frac{\log n}{2^j}}\right) & \text{if } j < \log\log n \text{ (light load)}, \\ O(2^j) & \text{if } j \geq \log\log n \text{ (heavy load)}. \end{cases}$

The recourse bound is the best possible, since the average load in the j-th phase is $\Theta(2^j)$, and there are $\frac{n}{2^j}$ rounds. The maximum load matches that of throwing $O(n)$ balls to $\frac{n}{2^j}$ bins, and thus is the best possible. The proof follows by summation over the ℓ phases, which might be slightly sub-optimal for bounding the maximum load when $\omega(1) \leq \ell \leq \log\log n + \log\log\log\log n$.

The proof uses the following two lemmas. The first reduces (via coupling) each phase j to the following non-adaptive game, and the second bounds the recourse and maximum load in this non-adaptive game. In the j-th non-adaptive game, balls are thrown one at a time to $n/2^{j-1}$ bins until every subset of $n/2^j$ bins contains at least n balls. We denote by Y_j the number of inserted balls, and by B_j the maximum load (which are random variables).

▶ **Lemma 3.3.** *Let $j \in [\log n]$. Fix an adaptive adversary, and denote its recourse in the j-th phase by X_j and the load incurred in the j-th phase by A_j. Then, $Y_j \succeq X_j$ and $B_j \succeq A_j$, where Y_j and B_j are the number of inserted balls and the maximum load in the j-th non-adaptive game defined above.*

Proof. We begin by analyzing the recourse. We construct a coupling between Y_j and X_j by simulating the non-adaptive game during the $n/2^j$ rounds of the adaptive game. Abusing notation slightly, we continue to use X_j and Y_j to denote these random variables within the coupled setting.

For the non-adaptive game, let L_i^{na} denote the load of the i-th bin for $i \in [n/2^{j-1}]$. In the adaptive game, denote by $\texttt{Alive}_j \subseteq [n]$ the set of living bins right before the j-th phase starts. In particular, $|\texttt{Alive}_j| = n/2^{j-1}$, and therefore there is a one-to-one mapping between $\texttt{Alive}_j$ and $[n/2^{j-1}]$. For simplicity of presentation, we omit this mapping and simply treat every $i \in \texttt{Alive}_j$ as a number in $[n/2^{j-1}]$. Let $\texttt{Alive} \subseteq \texttt{Alive}_j$ denote the set of living bins during phase j; for each $i \in \texttt{Alive}$, let L_i^a be the load of the i-th bin incurred by balls that move in the j-th phase. We maintain the following invariant:

- **Invariant.** For every $i \in \texttt{Alive}$, we have $L_i^{na} = L_i^a$.

Initially, in the non-adaptive game there are no balls. No balls moved yet in the adaptive game. Clearly, the invariant holds.

Now, consider a round of the adaptive game. The adaptive adversary picks a bin to delete, say containing b balls. Suppose these b balls are thrown one at a time in the adaptive game, and let us simulate throwing one ball using the non-adaptive game. In the non-adaptive game, we throw a ball uniformly over all the $n/2^{j-1}$ bins. If the ball is inserted to a bin in $[n/2^{j-1}] \setminus \texttt{Alive}$, i.e., bin that was deleted in the adaptive game, we repeatedly throw another new ball to the $n/2^{j-1}$ bins, until we get a ball in some bin $i \in \texttt{Alive}$. Therefore,

for all $i' \in \texttt{Alive}, i' \neq i$, we have that $L^{na}_{i'}$ is unchanged, and $L^{na}_i \leftarrow L^{na}_i + 1$. We copy that last ball to the adaptive game, i.e., we only update $L^a_i \leftarrow L^a_i + 1$, and so the invariant holds. This process is repeated for all the b balls, and then we proceed to the next round of the adaptive game.

Crucially, we claim that the above is a simulation of the adaptive game, and it may end only after the adaptive adversary deletes $n/2^j$ bins. In the non-adaptive game, the distribution of a thrown ball, given that it lands in $\texttt{Alive}$, is uniform over $\texttt{Alive}$. Thus, the above indeed simulates the adaptive game.

Observe that right before the adaptive adversary deletes the $\frac{n}{2^j}$-th bin, we have that $|\texttt{Alive}| = \frac{n}{2^j} + 1$. Since $\sum_{i \in \texttt{Alive}} L^a_i \leq n$, there must exists a subset of $\texttt{Alive}$ of size $\frac{n}{2^j}$, whose bins in the adaptive game contain strictly less than n balls. Therefore, by the invariant, also in the non-adaptive game, there is a subset of $\frac{n}{2^j}$ bins containing strictly less than n balls, and thus the non-adaptive game didn't end. Hence, the number Y_j of balls in the non-adaptive game is at least the recourse X_j of the adaptive game in the j-th phase, i.e., $Y_j \geq X_j$.

Over the same probability space, we immediately obtain a similar result for the maximum load. The maximum load B_j in the non-adaptive game is larger than the maximum load in $\texttt{Alive}$, which by the invariant, equals the maximum load incurred by balls that move in the j-th phase of the adaptive game, and thus $B_j \geq A_j$. The proof is concluded by Lemma 1.7. ◀

▶ **Lemma 3.4.** *Consider the j-th non-adaptive game and let Y_j be the number of balls it throws. Then, $Y_j \leq 10n$ with probability at least $1 - e^{-2n}$.*

Proof. Consider the classical balls into bins problem with $10n$ balls, and each ball is thrown uniformly to $n/2^{j-1}$ bins. We show that with high probability, all subsets of $n/2^j$ bins contain at least n balls, and the claim follows.

Let $S \subset [n/2^{j-1}]$ be a subset of size $|S| = \frac{n}{2^j}$. For every $i \in [10n]$, denote by I_i an indicator that the i-th ball is in S. Thus, the number of balls in S equals $\sum_{i=1}^{10n} I_i$. Clearly, $I_i = 1$ w.p. $\frac{1}{2}$ and 0 otherwise, and all these indicators are independent. Thus, by Hoeffding's inequality,

$$\Pr\Big(\sum_{i=1}^{10n} I_i < n\Big) \leq \Pr\Big(|\sum_{i=1}^{10n} I_i - 5n| > 4n\Big) \leq 2\exp\Big(-\frac{2(4n)^2}{10n}\Big) \leq 2\exp(-3n).$$

By a union bound, the probability that there exists a set $S \subset [n]$ of size $|S| = \frac{n}{2^j}$ containing less than n balls is at most

$$\binom{n/2^{j-1}}{n/2^j} \cdot 2\exp(-3n) \leq 2^n \cdot 2\exp(-3n) \leq \exp(-2n),$$

which concludes the proof of the lemma. ◀

Proof of Theorem 3.1. By combining Lemma 3.3 and Lemma 3.4, we have that for each $j \in [\ell]$, with probability at least $1 - e^{-2n}$, the recourse is at most $10n$. By a union bound, with probability at least $1 - e^{-n}$, the total recourse is $O(\ell n)$. In standard balls and bins with $10n$ balls and $n/2^{j-1}$ bins, with probability $1 - \frac{1}{n^{11}}$, the maximum load is [27]

$$\begin{cases} O\Big(\frac{\log n}{\log \frac{\log n}{2^j}}\Big) & \text{if } j < \log\log n, \\ O(2^j) & \text{if } j \geq \log\log n. \end{cases}$$

Thus, by a union bound, with probability $1-n^{-10}$, the maximum load B_j in the non-adaptive game is bounded by that as well. Observe that

$$\sum_{j=1}^{\ell} \frac{1}{\log \frac{\log n}{2^j}} = \sum_{x=\log\log n-\ell}^{\log\log n-1} \frac{1}{x} \leq \int_{\log\log n-\ell-1}^{\log\log n} \frac{dx}{x} = O(\log \frac{\log\log n}{\log\log n-\ell}).$$

When $\ell \leq \frac{1}{2}\log\log n$, we have that $\log \frac{\log\log n}{\log\log n-\ell} = \log(1+\frac{\ell}{\log\log n-\ell}) = O(\frac{\ell}{\log\log n})$. Thus, with probability $1-n^{-10}$, the maximum load in the adaptive game is

$$\leq \sum_{j=1}^{\ell} A_j \leq \sum_{j=1}^{\ell} B_j$$

$$\leq \begin{cases} O(\log n \log \frac{\log\log n}{\log\log n-\ell}) & \text{if } \ell < \log\log n, \\ O(2^\ell + \log n \log\log\log n) & \text{if } \ell \geq \log\log n, \end{cases}$$

$$= \begin{cases} O(\frac{\ell \log n}{\log\log n}) & \text{if } \ell \leq \frac{1}{2}\log\log n, \\ O(\log n \log \frac{\log\log n}{\log\log n-\ell}) & \text{if } \frac{1}{2}\log\log n < \ell < \log\log n, \\ O(\log n \log\log\log n) & \text{if } \log\log n \leq \ell < \log\log n + \log\log\log\log n, \\ O(2^\ell) & \text{if } \ell \geq \log\log n + \log\log\log\log n, \end{cases}$$

and the proof follows. ◀

3.1 Two choices

Consider the following adaptation of the game above: whenever a ball is thrown, we pick two independent bins b_1, b_2, and place the ball at the least loaded of b_1, b_2. This is applied both to the initial configuration, and to balls thrown during the game. This is based on the well-known power of two choices, which in standard (non-adaptive) balls and bins setting, yields a maximum load of $O(\log\log n)$ when $m = O(n)$ [3, 11, 12]. We show the following.

▶ **Theorem 3.5.** *Let $T \in [n-1]$ and denote $\ell = \lceil \log \frac{n}{n-T} \rceil$. In T rounds of adaptive two choices game, the recourse is $O(\ell n)$ and the maximum load is $O(2^\ell + \ell \log\log n)$, with high probability.*

The proof is similar to that of Theorem 3.1. We reduce the problem (via coupling) to a non-adaptive game, and then bound the parameters of this non-adaptive game. The non-adaptive game is similar, only now we consider pairs of indices. For simplicity, consider $n/2$ rounds (i.e., $\ell = 1$). In this new non-adaptive game, iteratively pick two indices independently and uniformly at random over $[n]$, place a ball in the least loaded of them, and stop when every subset of size $n/2$ contains at least n pairs. Denote by Y the number of inserted balls, and by B the maximum load (which are random variables). We show in Appendix A that Lemma 3.3 still holds in this setting, using the same coupling, and the number of rounds here is still $O(n)$ with high probability. Thus, with high probability, the recourse is $O(n)$. To bound the maximum load, recall that when throwing m balls into n bins using two choices allocations, if $m \geq n$, the maximum load is at most $\frac{m}{n} + O(\log\log n)$ with high probability [3, 11]. In our case, $m = 10n$, thus the maximum load is $O(\log\log n)$, yielding Theorem 3.5. A proof is provided in Appendix A.

4 d-split problem, $d = 2$

We now consider the 2-split problem, played for $n/2$ steps. That is, when a bin is deleted, the balls inside are partitioned into two uniformly random (remaining) bins. We show that in $\frac{n}{2}$ rounds, the recourse is $O(n)$ with high probability, and the maximum load is $\text{poly}(\log n)$ with constant probability. While the constant 2 in the number of rounds is arbitrary, bounding the recourse and maximum load when the number of rounds approaches n remains open.

4.1 Bounding the recourse

▶ **Theorem 4.1.** *In $\frac{n}{2}$ rounds of the 2-split game, the recourse is $O(n)$ with probability $1-\frac{1}{n^7}$.*

As we mentioned in the introduction, our proof uses an adaptive balls into bins game, denoted π_{pay}, where there is an extra $O(k \log k)$ cost for moving $k \geq 2$ balls from a bin with k balls, and the balls are thrown independently uniformly at random.

▶ **Lemma 4.2.** *In $n/2$ rounds of π_{pay}, the total cost is $O(n)$ with probability $1-\frac{1}{n^8}$.*

▶ **Lemma 4.3.** *The total cost in $n/2$ rounds of π_{pay} stochastically dominates a RV that is at total variation distance at most $\frac{1}{n^8}$ from the recourse in $n/2$ rounds of the 2-split game.*

Theorem 4.1 follows from these two lemmas and a union bound. We first prove Lemma 4.3. The proof of Lemma 4.2 is provided in Section 4.1.3.

4.1.1 Proof of Lemma 4.3

Recall that our proof uses a toy problem, as follows. There are $k \geq 2$ balls, initially located at the same bin. In rounds, the adversary deletes a bin that contains at least 2 balls, and the balls are split evenly to two sets that are thrown independently. The game ends when the adversary cannot delete bins, that is, every bin has at most one ball.

▶ **Lemma 4.4.** *If $k = O(\log n)$, then with probability $1-\frac{1}{n^{10}}$, the recourse in the toy problem is $O(k \log k)$.*

The proof of Lemma 4.4 is given in Section 4.1.2. We now prove Lemma 4.3 using Lemma 4.4.

Proof of Lemma 4.3. To connect 2-split to π_{pay}, both played for $n/2$ rounds, we consider a sequence of hybrid games, $\pi_0, \pi_1, \ldots, \pi_{n/2}$, where:

(1) Game π_0 is exactly the 2-split game.

(2) For every $t \in [n/2]$, with high probability, it holds that the cost of π_t is at least the cost of π_{t-1}.

(3) The cost of π_{pay} stochastically dominates the cost of $\pi_{n/2}$.

Recall that in the game π_{pay}, whenever a bin with k balls is deleted we incur an extra cost of $O(k \log k)$, and the balls are thrown independently uniformly at random. The sequence of hybrid games we consider $(\pi_0, \pi_1, \ldots, \pi_{n/2})$ gradually interpolate between the 2-split game and this π_{pay} game. Intuitively, but somewhat inaccurately, in the game π_t, the first t deletions behave like in π_{pay} (balls are thrown independently and our recourse has a logarithmic blowup), and the rest of the $\frac{n}{2}-t$ deletions behave like in the 2-split game (balls are partitioned into two groups). We now make this formal, starting with the formalization of the game π_1.

The game π_1. The game π_1 is identical to the 2-split game, with one key modification regarding the balls from the *first* bin that is deleted. Suppose that the first deletion is of a bin with k balls. We pay an extra cost of $O(k \log k)$ and redistribute the k balls *independently and uniformly at random*. From now on, these k balls are special; we call these k balls "Teflon" and treat them differently throughout the rest of the game π_1. Unlike normal balls, Teflon balls do not "stick". That is, throughout the rest of the game π_1, when a bin is deleted, if this bin contains Teflon balls, then these Teflon balls are thrown independently and uniformly at random into the remaining bins. Note that "normal" balls from the deleted bin are still split into two sets and thrown as groups. Teflon balls may eventually revert to being "normal" balls based on coupling conditions we describe next between the executions of π_1 and the 2-split game.

The coupling between 2-split and π_1. Fix an adversary $\mathcal{A}$ for playing in the 2-split game (i.e., in π_0). We consider a coupled execution of 2-split and π_1 defined as follows. The initial configuration is identical in both games, obtained by throwing n balls uniformly into the n bins. The bin deletion sequence is kept identical in both games, and is determined by $\mathcal{A}$ based on the configuration of the balls in the 2-split game ($\mathcal{A}$ does not see the configuration of the balls in π_1). The first deletion made by $\mathcal{A}$ defines the identity of the Teflon balls in π_1. We stress that in the 2-split game, Teflon balls behave as any other ball. To emphasize this, whenever we refer to these balls in the 2-split game we will call them the "so-called Teflon balls". Throughout the execution, whenever a bin is deleted, we redistribute its balls as follows in each of the games:

- In both games, normal balls are split evenly and thrown to the same randomly chosen bin index.
- In π_1, Teflon balls are thrown uniformly and independently.
- In the 2-split game, Teflon balls from the deleted bin are split evenly to two sets and attached to the two sets of the normal balls. The first time when the bin of so-called Teflon-ball b is deleted in the 2-split game, and b is assigned to a set that includes no other so-called Teflon-balls, we declare b as normal (not Teflon anymore in both games), and throw its set to the bin of the same index as the bin containing b in π_1. That is, the randomness used to pick the identity of the new bin is taken from π_1. Since b was previously thrown independently in π_1, then from the viewpoint of the 2-split game, the thrown set is distributed independently, as desired.

Let X denote the total recourse that $\mathcal{A}$ achieves in $n/2$ rounds of the 2-split game, and let Y_1 denote the incurred cost in the coupled execution of π_1. We show that with high probability, $Y_1 \geq X$.

▷ Claim 4.5. With probability $1 - \frac{1}{n^9}$, $Y_1 \geq X$.

Proof. By the invariant, the recourse of the normal balls is the same. We ignore the contribution to Y_1 from the additional recourse the Teflon-balls incur in π_1. Thus, we only have to bound the recourse of the so-called Teflon balls in the 2-split game until they become normal, and compare it to the extra $O(k \log k)$ cost paid in π_1 at the first round. By standard arguments, with probability $1 - \frac{1}{n^{10}}$, the maximum load in the initial distribution of n balls into n bins is $O(\log n)$, and thus $k \leq O(\log n)$. Observe that in the special case where $k = 1$, the two games are identical, hence we consider $k \geq 2$. The recourse of the so-called Teflon-balls in the 2-split game is stochastically dominated by the recourse in the toy problem, which by Lemma 4.4 is $O(k \log k)$ with probability $1 - \frac{1}{n^{10}}$. By a union bound, we have $Y_1 \geq X$ with probability $1 - \frac{2}{n^{10}}$, which concludes the proof. ◁

The sequence of coupled games. We now inductively define the sequence of coupled games $\pi_2, \ldots, \pi_{n/2}$, where each game π_i introduces an additional set of Teflon balls, denoted T_i (where T_1 is the set of Teflon balls w.r.t. π_1 and where $T_0 = \emptyset$). For $t \geq 2$, we define the game π_t, and couple it with π_{t-1}, similar to the construction above. The two games π_{t-1} and π_t are played the same for $t-1$ rounds. In round t, we define T_t to be the set of all balls from the deleted bin that are not in $\cup_{i=1}^{t-1} T_i$. That is, T_t contains all normal balls (w.r.t. π_{t-1}) that are deleted in round t. Now, from round t onward:

- In π_{t-1}, balls that are not in $\cup_{i=1}^{t-1} T_i$ (aka normal w.r.t. π_{t-1}) are played as in 2-split, and balls in $\cup_{i=1}^{t-1} T_i$ (aka Teflon) are played as independent. Some Teflon-balls may become normal, depending on the coupling with π_i, for $i \leq t-2$.
- The only difference between π_{t-1} and π_t is about the balls in T_t, where in π_t these are played as Teflon-balls, and have extra cost $O(|T_t| \log |T_t|)$.
- In π_{t-1}, whenever a bin containing T_t-balls is deleted, we split the T_t-balls in that bin between the two sets. When a T_t-ball b is assigned to a set that contains no other T_t-ball in π_{t-1}, we throw its set to the bin of the same index as the bin containing b in π_t, and remove b from T_t. As before, from the viewpoint of π_{t-1}, this assignment is independent.

▷ Claim 4.6. Let $t \in [n/2]$. With probability $1 - \frac{1}{n^9}$, $Y_t \geq Y_{t-1}$.

Proof. This is a generalization of Claim 4.5, and the proof is similar. The only difference between Y_{t-1} and Y_t is in the cost and recourse caused by the Teflon balls T_t. We ignore the contribution to Y_t from the additional recourse the Teflon balls in T_t incur in π_t. Thus, we only compare the recourse of the balls in T_t in π_{t-1} until they become normal, with the cost $O(|T_t| \log |T_t|)$ paid in π_t.

First, observe that with probability $1 - \frac{1}{n^{10}}$, we have $|T_t| \leq O(\log n)$, by the following. For every $i \leq t-1$, the balls in a deleted bin in π_{t-1} are distributed independently. Therefore, the maximum load before round t, and hence $|T_t|$, is stochastically dominated by the maximum load in uniform adaptive balls into bins played for $n/2$ rounds. Hence, by Theorem 3.1, $|T_t| \leq O(\log n)$ with probability $1 - \frac{1}{n^{10}}$.

The recourse in π_{t-1} of the T_t-balls until they become normal, is stochastically dominated by the toy problem, with $k = |T_t|$. By Lemma 4.4 and a union bound, with probability $1 - \frac{1}{n^9}$, the recourse in π_{t-1} of the T_t-balls until they become normal is at most $O(|T_t| \log |T_t|)$. Under this same event, we have $Y_t \geq Y_{t-1}$, concluding the proof of the claim. ◁

To conclude the proof of Lemma 4.3, observe that the cost of π_{pay} stochastically dominates the cost of $\pi_{n/2}$, since the payments are only larger. (Because in $\pi_{n/2}$ we do not necessarily incur the logarithmic blowup in the cost on all the balls from a deleted bin.) By a union bound, we have that with probability $1 - \frac{1}{n^8}$, the sequence Y_t is non-decreasing, and thus the cost of π_{pay} is at least that of 2-split. ◀

4.1.2 Proof of Lemma 4.4, bounding the recourse in the toy problem

Proof of Lemma 4.4. Recall that in the toy problem, the initial configuration is that we have $2 \leq k \leq O(\log n)$ balls in one bin, and the other bins are empty. Then, in rounds, the adversary deletes a bin containing at least 2 balls, and the balls are split evenly to two sets that are thrown independently. The game ends when all bins contain at most 1 ball.

Towards analyzing the recourse, let us consider a modified variant of this toy problem, which we call π', where the adversary has the additional power to determine the initial configuration for the k balls. Furthermore, whenever a collision happens (i.e., whenever we throw a ball into a non-empty bin), the adversary can override the current configuration

and completely rearrange the k balls in the living bins. Clearly, the cost in π' stochastically dominates the recourse in the toy problem, so it suffices to bound the cost in π'. To this end, we express π' as a sequence of "mini-games" defined as follows:

Mini-game

In a mini-game, the adversary initially picks a configuration for the k balls in the living bins. Then, in every round, the adversary deletes a bin that contains at least 2 balls, and the balls are split evenly to two sets that are thrown independently. Now, if one of these sets is thrown to a non-empty bin (i.e., there is a collision), we conclude the current mini-game and start a new one. Otherwise, we play until all bins contain at most 1 ball, at which point we conclude the current mini-game without starting a new one (this corresponds to the end of π').

It is easy to see that a mini-game takes at most $O(k)$ rounds, and the actual recourse in a mini-game is bounded by $O(k \log k)$. Thus, to bound the total cost in π', it suffices to show that the number of mini-games is small with high probability. That is, we want to show that the probability that a mini-game ends (i.e., there is a collision) without π' ending is small.

Recall we have at least $n/2$ bins. Consider a round in a mini-game where the k balls are located in i bins. The adversary now deletes one of these bins, and the probability that one of the two sets we throw collide with a non-empty bin is at most $\frac{2i}{n/2} = O(\frac{k}{n})$. As there are at most $O(k)$ rounds in the mini-game, no matter what is the distribution that the adversary picked at the start of the mini-game, the probability of a collision is at most $O(\frac{k^2}{n})$.

The total cost in π' is the number of mini-games with a collision, times the cost $O(k \log k)$ of a mini-game. The number of mini-games is stochastically dominated by a geometric random variable, with success when there are no collisions, i.e., it has parameter $p = 1 - O(\frac{k^2}{n})$. Thus, the probability that there are more than 20 mini-games is at most $O(\frac{k^2}{n})^{20} \leq O(n^{-10})$. When this event does not happen, the total cost in π' is $O(k \log k)$. This concludes the proof. ◀

4.1.3 Proof of Lemma 4.2

Recall that in π_{pay}, the cost of deleting a bin with k balls is $k + O(k \log k)$, and the balls are thrown independently. Lemma 4.2 states that the cost of $n/2$ rounds of π_{pay} is $O(n)$ with high probability.

Proof of Lemma 4.2. The proof uses the same probability space as in the proof of Theorem 3.1. We consider a non-adaptive game, where balls are thrown into n bins until every subset of $n/2$ bins contains at least n balls. By the coupling in the proof of Theorem 3.1, we have a matching between deleted bin i in the adaptive game and bin i in the non-adaptive game, such that $L_i^{na} \geq L_i^a$, i.e., the load in the non-adaptive game is at least as in the adaptive game. Consider the $n/2$ heaviest loaded bins in the non-adaptive game, with loads $k_1, \ldots, k_{n/2}$. Take as cost $O\big(\sum_{i=1}^{n/2} k_i(1 + \log k_i)\big)$. By the coupling, this sum stochastically dominates the cost in π_{pay}. By Lemma 3.4, with high probability, the number of thrown balls in the non-adaptive game is at most $10n$. We now bound $\sum_{i=1}^{n/2} k_i \log k_i$ after throwing $10n$ balls, using the following.

▶ **Lemma 4.7.** *Consider standard balls into bins, with $10n$ balls and n bins. Denote the loads of the bins by $L_1, L_2, \ldots, L_n$. With high probability, $\sum_{i=1}^{n} L_i \log L_i = O(n)$.*

We defer the proof of Lemma 4.7 to Appendix B. By a union bound, after throwing $10n$ balls, the game ends and the cost is $O(n)$, which concludes the proof of Lemma 4.2. ◀

4.2 Bounding the maximum load

▶ **Theorem 4.8.** *The maximum load in the 2-split problem is* $\mathrm{poly}(\log n)$ *with constant probability. Moreover, for success probability* $1-\delta$*, the maximum load is at most* $\log^{O(\log \frac{1}{\delta})} n$*.*

Our proof uses the following lemma.

▶ **Lemma 4.9.** *Consider any assignment of* n *balls to* $m \in [n/2, n]$ *bins, with maximum load* L*. There is a constant* $C > 1$*, such that with probability* $1/2$*, the number of rounds of 2-split until the maximum load reaches* $O(L \cdot \log n)$ *is at least* $\min\{\frac{n}{C}, m - \frac{n}{2}\}$*.*

The proof of Lemma 4.9 uses the following simple claim.

▷ Claim 4.10. Let $w, n > 0$. For every $x \geq 0$, we have $(w - x^2)(1 + \frac{4x}{n}) \leq w(1 + \frac{4w}{n^2})$.

Proof. The difference satisfies

$$\begin{aligned} &w(1 + \tfrac{4w}{n^2}) - (w - x^2)(1 + \tfrac{4x}{n}) \\ &= w + \tfrac{4w^2}{n^2} - w - \tfrac{4wx}{n} + x^2 + \tfrac{4x^3}{n} \\ &= \tfrac{4w^2}{n^2} - \tfrac{4wx}{n} + x^2 + \tfrac{4x^3}{n} \\ &= (\tfrac{2w}{n} - x)^2 + \tfrac{4x^3}{n} \geq 0. \end{aligned}$$

◁

Proof of Lemma 4.9. Pick $\alpha = 1/L$. Denote the set of surviving bins at time $t \in [0, m - \frac{n}{2}]$ by $I_t \subseteq [m]$, and for every $i \in I_t$, denote the load of the i-th bin as L_i^t. When clear from context, we simply write L_i. Consider the potential function

$$w_t = \sum_{i \in I_t} e^{\alpha L_i}.$$

Initially, we have

$$w_0 = \sum_{i \in I_0} e^{\alpha L_i} \leq en.$$

Without loss of generality we may suppose that the adversary is deterministic. That is, in every step, given the current constellation H, the adversary deterministically chooses the next bin to eliminate (denoted as r). The only randomness is in picking the indices of the 2 bins to which we split r, denoted as x and y. For simplicity, we assume the algorithm enforces $x \neq y$. So going from time $t \leq m - \frac{n}{2}$ to time $t+1$, we have that w_t changes to w_{t+1} as follows:

$$\begin{aligned} w_{t+1} &= w_t - e^{\alpha L_r} + e^{\alpha L_x}\left(e^{\alpha \lceil L_r/2 \rceil} - 1\right) + e^{\alpha L_y}\left(e^{\alpha \lfloor L_r/2 \rfloor} - 1\right) \\ &\leq w_t - e^{\alpha L_r} + \left(2e^{\alpha L_r/2} - 1\right)\left[e^{\alpha L_x} + e^{\alpha L_y}\right]. \end{aligned}$$

We want to compute the expectation of this given the history H of the game until time t. The expectation is only over the choice of x, y.

$$\begin{aligned}
\mathbb{E}[w_{t+1}|H] &\leq w_t - e^{\alpha L_r} + \left(2e^{\alpha L_r/2} - 1\right) 2\mathbb{E}[e^{\alpha L_x}] \\
&\leq w_t - e^{\alpha L_r} + \left(2e^{\alpha L_r/2} - 1\right) 2 \sum_{i \in I_{t+1}} \tfrac{1}{|I_{t+1}|} e^{\alpha L_i} \\
&\underset{|I_{t+1}| \geq n/2}{\leq} w_t - e^{\alpha L_r} + \frac{2e^{\alpha L_r/2} - 1}{n/4} \left[w_t - e^{\alpha L_r}\right] \\
&= \left[w_t - e^{\alpha L_r}\right] \left(1 + \frac{2e^{\alpha L_r/2} - 1}{n/4}\right) \\
&\leq \left[w_t - e^{\alpha L_r}\right] \left(1 + \frac{2e^{\alpha L_r/2}}{n/4}\right) \\
&\leq w_t \left(1 + \frac{16 w_t}{n^2}\right),
\end{aligned}$$

The last inequality holds by Claim 4.10.

Let C be some large constant and consider the alternative random variables with an appropriate random stopping time τ defined as follows:

$$\tau = \max\{t : w_t \leq Cn\}, \qquad \text{and} \qquad \hat{w}_t = w_{t \wedge \tau},$$

where $t \wedge \tau = \min\{t, \tau\}$. Therefore,

$$\mathbb{E}[\hat{w}_{t+1}|H] \leq \hat{w}_t(1 + \tfrac{16\hat{w}_t}{n^2}) \leq \hat{w}_t(1 + \tfrac{16C}{n}),$$

and so, by the law of total expectation and since $w_0 \leq en$,

$$E[\hat{w}_{n/C}] \leq en \cdot \left(1 + \frac{16C}{n}\right)^{n/C} \leq en \cdot e^{16}.$$

By Markov's inequality,

$$\Pr[\hat{w}_{n/C} > 2e^{17} n] \leq \frac{1}{2}.$$

Set $C > 2e^{17}$. Thus, with probability at least $1/2$ we have that $\hat{w}_{n/C} \leq Cn$ in which case the stopping condition does not happen until step n/C, and so $w_{n/C} = \hat{w}_{n/C} \leq Cn$ and $\tau \geq n/C$. Observe that for every $t \leq \tau$, the maximum load is at most,

$$\tfrac{1}{\alpha} \log w_t \leq L \log(Cn),$$

which concludes the proof of Lemma 4.9. ◀

Before proceeding to the proof of Theorem 4.8, we recall useful probability statements regarding martingales (and particularly submartingales).

▶ **Definition 4.11.** *A sequence $X_1, X_2, \ldots$ of random variables with finite mean, is called submartingale if $\mathbb{E}[X_{i+1}|X_i, \ldots, X_1] \geq X_i$, for all $i \geq 1$; is called supermartingale if $\mathbb{E}[X_{i+1}|X_i, \ldots, X_1] \leq X_i$, for all $i \geq 1$; and is called martingale if it is both submartingale and supermartingale.*

We will use the following (special case of) Azuma's inequality for submartingales.

▶ **Theorem 4.12** (Azuma's inequality for submartingales). *Suppose $X_1, X_2, \ldots$ is a submartingale and $|X_{i+1} - X_i| \le c$. Then, for all $n \in \mathbb{N}$ and $t > 0$,*

$$\Pr(X_n - X_1 \le -t) \le \exp\Big(-\frac{t^2}{2nc^2}\Big).$$

Proof of Theorem 4.8. By standard arguments, the initial maximum load is $\le \log n$ with high probability (say $\ge 1 - n^{-10}$ for concreteness). Assume henceforth that this event holds. Denote by τ_i the number of rounds until the maximum load reaches $\log n \cdot \log^i Cn = O(\log^{i+1} n)$ where C is a constant satisfying the requirements of Lemma 4.9. It follows that $\tau_0 = 0$. Define the index $t^* = \max_{\tau_i \le n/2} i$, i.e., the index of the last τ_i before the $\frac{n}{2}$-th round of the game. Given the constellation at time τ_i, for $i \le t^*$, by Lemma 4.9, $\tau_i - \tau_{i-1} \ge n/C$ w.p. at least $1/2$. Denote by I_i the indicator that $\tau_i - \tau_{i-1} \ge n/C$, and $S_t = \sum_{i=1}^{t \wedge t^*} I_i - \frac{t}{2}$. We have that

$$\mathbb{E}[S_t \mid S_{t-1}, \ldots, S_1] = \begin{cases} S_{t-1} + \mathbb{E}\left[I_t - \frac{1}{2} \mid S_{t-1}, \ldots, S_1\right], & \text{if } t \le t^* \\ S_{t-1}, & \text{otherwise} \end{cases}$$
$$\ge S_{t-1}.$$

Thus, S_t is a submartingale. By Azuma's inequality,

$$\Pr[S_t \le -\tfrac{t}{4}] \le \exp(-\frac{t}{32}).$$

Assume this event does not hold.

Now, assume by contradiction that $2C < t^*$. We obtain $\sum_{i=1}^{2C} I_i = S_{2C} + \frac{2C}{2} \ge \frac{2C}{4} = \frac{C}{2}$. Thus, at least $\frac{C}{2}$ of the indicators $\{I_i\}_{i=1}^{2C}$ are equal 1, and thus $\tau_{2C} \ge \frac{C}{2} \cdot \frac{n}{C} = \frac{n}{2}$. Therefore, $2C \ge t^*$, contradiction. We thus obtained that $t^* \le 2C$. Therefore, the maximum load is $O(\log^{2C+1} n)$ with probability at least $1 - \exp(-\frac{C}{16}) - n^{-10}$. Setting $C = \max\{C' \log(1/\delta), 2e^5\}$ for an appropriate constant C' concludes the proof. ◀

5 d-split problem, $d = 1$, oblivious adversary

We now consider the d-split problem, with $d = 1$, i.e., all the balls in a deleted bin move to the same new bin; and play the game for $n/2$ steps. With adaptive adversary, this strategy is clearly bad, as the adversary can delete the heaviest bin and create a snowball effect. For example, in the initial configuration where every bin contains exactly one ball, the recourse is $\Omega(n^2)$ and the maximum load is $n/2$ (and similar bounds seem to hold for other initial configurations). Nevertheless, we establish that this strategy is solid against oblivious adversaries. Suppose the initial configuration has one ball in each bin. For simplicity of presentation, we call this variant oblivious 1-split.

▶ **Theorem 5.1.** *Consider oblivious 1-split, i.e., where all balls in a deleted bin are thrown together to a single new bin, the adversary is oblivious, and the initial configuration where every bin contains exactly one ball. After $n/2$ steps, the total recourse is $O(n \log n)$ and the max-load is $O(\log n)$, with high probability.*

Recall the intuition behind this result. Consider a random *forest* graph on n vertices constructed as follows: for each step i from 1 to $n/2$, vertex i uniformly samples a parent $j \in \{i+1, \ldots, n\}$ and becomes a child of j. This process yields $n/2$ trees $T_1, \ldots, T_{\frac{n}{2}}$, whose roots are the vertices indexed $n/2 + 1$ through n. As explained right after Theorem 1.5, the maximum load in oblivious 1-split equals the size of the largest tree in this forest, and the recourse equals the sum of depths of all vertices in this forest.

5.1 Bounding the recourse

We first bound the recourse of oblivious 1-split by $O(n \log n)$ with high probability. Recall that Y is a geometric RV with parameter $\frac{1}{2}$ if $\Pr(Y = j) = 2^{-j}$ for every integer $j \geq 1$.

▷ Claim 5.2 (Lemma 5 of [16]). For every $i \leq n/2$, the recourse of the ball that started in the i-th bin is stochastically dominated by a geometric random variable with parameter $\frac{1}{2}$.

Therefore, for each $i \in [n]$, with probability $1 - n^{-10}$, the recourse of the i-th ball is $O(\log n)$. By a union bound, with probability $1 - n^{-9}$, the recourse of the i-th ball is $O(\log n)$ for all $i \in [n/2]$. Thus, the overall recourse is $O(n \log n)$ with probability $1 - n^{-9}$.

5.2 Bounding the max-load

We now show that the size of each of the trees $T_1, \ldots, T_{\frac{n}{2}}$ is stochastically dominated by a Galton-Watson Process (aka branching process), which yields the $O(\log n)$ bound on the maximum tree size.

▶ **Definition 5.3** (Galton-Watson Process). *A stochastic process $\{X_i\}_{i \in \mathbb{N}}$ is called Galton-Watson (GW) if it follows a recurrence formula $X_1 = 1$ and $X_{i+1} = \sum_{j=1}^{X_i} Y_j^i$, where Y_j^i are iid natural numbered RVs. The distribution μ of the Y_j^i is called the offspring distribution. The total progeny of the process is $\sum_{i=1}^{\infty} X_i$.*

Consider a random variable that is a sum of $n/2$ independent Bernoulli random variables, with success probabilities $\frac{1}{n/2}, \frac{1}{n/2+1}, \ldots, \frac{1}{n-1}$, and denote its distribution by μ.

▶ **Lemma 5.4.** *Let T be one of the trees defined above. The size of T is stochastically dominated by the total progeny of a Galton-Watson Process $\{X_i\}_{i \in \mathbb{N}}$ with offspring distribution μ.*

Proof. Without loss of generality, let T be the random tree rooted at $n/2 + 1$, where every node $i \leq n/2$ chooses a parent $parent(i)$ uniformly and independently from $\{i + 1, \ldots, n\}$. We introduce a coupling of the size of T to the total progeny of a GW Process $\{X_i\}_{i \in \mathbb{N}}$ with offspring distribution μ. Observe that a GW process can be defined by a tree T^{GW} s.t. the number of nodes in level i is X_i, and the number of children of the j-th node in level i is Y_j^i. The total progeny of the GW process is the size of T^{GW}.

In our coupling, we enforce $T \subseteq T^{GW}$ by making each node in T produce fewer or equal children than its counterpart in T_{GW} under this coupling. The number of children of the roots of T and T^{GW} are distributed according to μ, so coupling them is trivial (we use the same random coins).

Traverse T from right to left, and denote the subtree we observed so far by T_{now}. That is, initially T_{now} is just the root node $n/2 + 1$. In an inductive manner, suppose T_{now} is a subtree also of T^{GW}. We now add new nodes of T to T_{now}, and match them with nodes of T^{GW}. Denote by i the leaf of T_{now} with the maximum index. Every children u of i is chosen only from available indices (i.e., $u \notin T_{now}$), and must satisfy $u < i$. We denote the set of potential children of i by U_{now}, i.e., $U_{now} = \{u < i : u \notin T_{now}\}$. By construction, the set of potential parents of a node $u \in U_{now}$ equals $[u + 1, i] \cup ([i + 1, n] \setminus T_{now})$, i.e., vertices in $[u + 1, n]$ that are not internal nodes of T_{now} (since i is the leaf of T_{now} with the maximum index). Denote $P = [i, \frac{n}{2}] \setminus T_{now}$.

Given T_{now}, the number of children of i is a sum of $|U_{now}|$ independent Bernoulli random variables with success probabilities

$$\frac{1}{\frac{n}{2} + |P| + i - u}, \qquad u \in U_{now}.$$

Recall that the offspring distribution μ in T^{GW} is defined by a sum of independent Bernoulli's with success probabilities $\frac{1}{\frac{n}{2}+j}$ for $j \in [0, n/2-1]$. Observe that $|P| \leq \frac{n}{2} - i$. Thus, we can match the $|U_{now}|$ Bernoulli variables describing i's children to the corresponding Bernoulli variables in T^{GW} that have identical success probabilities. This forces the number of children of i in T to be stochastically bounded by the number of children of i in T^{GW}. The remaining Bernoulli variables in T^{GW} simply utilize another independent source of randomness. This concludes the inductive step.

We thus obtained a coupling between T and T^{GW}, s.t. $T \subseteq T^{GW}$. Therefore, $|T| \leq |T^{GW}|$, concluding the proof. ◀

We now bound the size of T^{GW} (or equivalently, the total progeny of the GW process) using Bernstein's inequality (Lemma 2.2). We first bound the expectation and variance of $Y \sim \mu$.

▶ **Lemma 5.5.** *Let $Y \sim \mu$. The following hold.*

- $\mathbb{E}Y < 0.7$*, and*
- $\operatorname{var} Y < 0.7$.

To prove Lemma 5.5, we use the following claim.

▷ Claim 5.6.

$$\sum_{i=1}^{n/2} \tfrac{1}{n-i} \leq \ln 2 + O(\tfrac{1}{n}).$$

Proof. Denote the n-th harmonic number by $H_n = \sum_{k=1}^{n} \frac{1}{k}$. Using the asymptotic expansion $H_n = \ln n + \gamma + \frac{1}{2n} + O(n^{-2})$ (where γ is the Euler-Mascheroni constant),

$$H_{n-1} - H_{n/2-1} = (\ln n + \gamma) - (\ln(n/2) + \gamma) + O(\tfrac{1}{n}) = \ln 2 + O(\tfrac{1}{n}).$$ ◁

Proof of Lemma 5.5. We have, $\mathbb{E}Y = \sum_{i=1}^{n/2} \frac{1}{n-i}$. By Claim 5.6, $\mathbb{E}Y \leq \ln 2 + O(\frac{1}{n})$. As $\ln 2 \approx 0.693$, this concludes the proof of the first bullet. Denote by I_i a Bernoulli random variable with success probability $\frac{1}{n+i}$, for $i = 1, \ldots, n/2$. We have

$$\operatorname{var} Y = \sum_{i=1}^{n/2} \operatorname{var} I_i \leq \sum_{i=1}^{n/2} \mathbb{E}I_i^2 = \sum_{i=1}^{n/2} \mathbb{E}I_i = \mathbb{E}Y < 0.7,$$

which concludes the proof. ◀

Let us now bound the total progeny of the GW process.

▶ **Lemma 5.7.** *With probability $1 - n^{-10}$, the total progeny of a GW process with offspring distribution μ is $O(\log n)$.*

Proof. Enumerate the nodes of the tree using a search order (e.g., Breadth-First Search). Let ξ_i be the number of children of the i-th node visited. Since nodes reproduce independently, the sequence $\xi_1, \xi_2, \ldots$ consists of i.i.d. random variables distributed according to μ. Let S_k be the number of unexplored nodes after visiting k nodes. We have the following recurrence: Start with one root, $S_0 = 1$, and at each step i, remove the current node and add its children

$$S_i = S_{i-1} - 1 + \xi_i.$$

Thus,

$$S_k = 1 + \sum_{i=1}^{k} (\xi_i - 1).$$

The process stops when there are no nodes to explore ($S_k = 0$). The total progeny Z is exactly the smallest index k such that $S_k = 0$. Therefore, the event that the tree size is large ($Z \geq k$) implies that $S_i \geq 1$ for all $0 \leq i \leq k$. A loose but sufficient condition for $Z \geq k$ is that the sum of offspring is at least the number of parents processed (minus the initial root),

$$Z \geq k \implies \sum_{i=1}^{k} \xi_i \geq k - 1.$$

Recall that each ξ_i is a sum of $\frac{n}{2}$ Bernoulli random variables (with different distributions), thus $\sum_{i=1}^{k} \xi_i$ is in fact a sum of $kn/2$ Bernoulli random variables. Hence, by Lemma 5.5 and Bernstein's bound,

$$\Pr(\sum_{i=1}^{k} \xi_i \geq k - 1) \leq 2\exp(-\tfrac{(0.3k)^2}{0.7k+0.3k/3}).$$

Taking $k = O(\log n)$, we obtain that the total progeny $Z \leq k = O(\log n)$ w.p. $1 - n^{-10}$, as desired. ◀

We are now ready to bound the max-load in Theorem 5.1, and thus conclude its proof.

Proof of Theorem 5.1, max-load. By Lemma 5.4, the load of a bin is dominated by the size of a GW process with offspring distribution μ. By Lemma 5.7, its size is $O(\log n)$ with probability $1 - n^{-10}$. A union bound over the $n/2$ trees concludes the proof. ◀

References

1 Noga Alon, Omri Ben-Eliezer, Yuval Dagan, Shay Moran, Moni Naor, and Eylon Yogev. Adversarial laws of large numbers and optimal regret in online classification. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 447–455, 2021. doi:10.1145/3406325.3451041.

2 Yossi Azar, Andrei Z. Broder, and Anna R. Karlin. On-line load balancing. *Theoretical Computer Science*, 130(1):73–84, 1994. doi:10.1016/0304-3975(94)90153-8.

3 Yossi Azar, Andrei Z. Broder, Anna R. Karlin, and Eli Upfal. Balanced allocations. *SIAM Journal on Computing*, 29(1):180–200, 1999. doi:10.1137/S0097539795288490.

4 Nikhil Bansal and William Kuszmaul. Balanced Allocations: The Heavily Loaded Case with Deletions . In *Proceedings of the 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 801–812, 2022. doi:10.1109/FOCS54457.2022.00081.

5 Luca Becchetti, Andrea Clementi, Emanuele Natale, Francesco Pasquale, and Gustavo Posta. Self-stabilizing repeated balls-into-bins. *Distributed Comput.*, 32(1):59–68, 2019. doi:10.1007/S00446-017-0320-4.

6 Amos Beimel, Haim Kaplan, Yishay Mansour, Kobbi Nissim, Thatchaphol Saranurak, and Uri Stemmer. Dynamic algorithms against an adaptive adversary: generic constructions and lower bounds. In *54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1671–1684. ACM, 2022. doi:10.1145/3519935.3520064.

7 Omri Ben-Eliezer, Rajesh Jayaram, David P. Woodruff, and Eylon Yogev. A framework for adversarially robust streaming algorithms. *J. ACM*, 69(2):17:1–17:33, 2022. doi:10.1145/3498334.

8 Omri Ben-Eliezer and Eylon Yogev. The adversarial robustness of sampling. In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS*, pages 49–62. ACM, 2020. doi:10.1145/3375395.3387643.

9 Michael A. Bender, Jake Christensen, Alex Conway, Martin Farach-Colton, Rob Johnson, and Meng-Tsung Tsai. Optimal ball recycling. In *Proceedings of the 30th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2527–2546, 2019. doi:10.1137/1.9781611975482.155.

10 Petra Berenbrink, André Brinkmann, Tom Friedetzky, and Lars Nagel. Balls into non-uniform bins. *Journal of Parallel and Distributed Computing*, 74(2):2065–2076, 2014. doi:10.1016/j.jpdc.2013.10.008.

11 Petra Berenbrink, Artur Czumaj, Angelika Steger, and Berthold Vöcking. Balanced allocations: the heavily loaded case. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 745–754, 2000. doi:10.1145/335305.335411.

12 Petra Berenbrink, Artur Czumaj, Angelika Steger, and Berthold Vöcking. Balanced allocations: The heavily loaded case. *SIAM Journal on Computing*, 35(6):1350–1385, 2006. doi:10.1137/S009753970444435X.

13 Petra Berenbrink, Tom Friedetzky, Zengjian Hu, and Russell Martin. On weighted balls-into-bins games. *Theoretical Computer Science*, 409(3):511–520, 2008. doi:10.1016/j.tcs.2008.09.023.

14 Aaron Bernstein, Jan van den Brand, Maximilian Probst Gutenberg, Danupon Nanongkai, Thatchaphol Saranurak, Aaron Sidford, and He Sun. Fully-dynamic graph sparsifiers against an adaptive adversary. In *49th International Colloquium on Automata, Languages, and Programming, ICALP*, LIPIcs, pages 20:1–20:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ICALP.2022.20.

15 Richard Cole, Alan Frieze, Bruce M Maggs, Michael Mitzenmacher, Andréa W Richa, Ramesh Sitaraman, and Eli Upfal. On balls and bins with deletions. In *International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 145–158, 1998. doi:10.1007/3-540-49543-6_12.

16 Adi Fine, Haim Kaplan, and Uri Stemmer. Minimizing recourse in an adaptive balls and bins game. In *Proceedings of the 52nd International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 334 of *LIPIcs*, pages 77:1–77:19, 2025. doi:10.4230/LIPIcs.ICALP.2025.77.

17 Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. In *Proceedings of the Conference on Analysis of Algorithms, AofA 07*, pages 127–146. DMTCS Proceedings, 2007.

18 P. Brighten Godfrey. Balls and bins with structure: balanced allocations on hypergraphs. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 511–517, 2008. URL: http://dl.acm.org/citation.cfm?id=1347082.1347138.

19 Gaston H. Gonnet. Expected length of the longest probe sequence in hash code searching. *J. ACM*, 28(2):289–304, 1981. doi:10.1145/322248.322254.

20 Bernhard Haeupler, Marc Kaufmann, Raghu Raman Ravi, and Ulysse Schaller. Adversarially-robust gossip algorithms for approximate quantile and mean computations. In *17th Innovations in Theoretical Computer Science Conference, ITCS*, LIPIcs, pages 74:1–74:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. doi:10.4230/LIPIcs.ITCS.2026.74.

21 Avinatan Hassidim, Haim Kaplan, Yishay Mansour, Yossi Matias, and Uri Stemmer. Adversarially robust streaming algorithms via differential privacy. *J. ACM*, 69(6):42:1–42:14, 2022. doi:10.1145/3556972.

22 David Karger, Eric Lehman, Tom Leighton, Rina Panigrahy, Matthew Levine, and Daniel Lewin. Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the world wide web. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC)*, pages 654–663, 1997. doi:10.1145/258533.258660.

23 Krishnaram Kenthapadi and Rina Panigrahy. Balanced allocation on graphs. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithm (SODA)*, pages 434–443, 2006. URL: http://dl.acm.org/citation.cfm?id=1109557.1109606.

24 Dimitrios Los and Thomas Sauerwald. Balanced allocations with incomplete information: The power of two queries. In *The 13th Innovations in Theoretical Computer Science (ITCS)*, volume 215 of *LIPIcs*, pages 103:1–103:23, 2022. doi:10.4230/LIPIcs.ITCS.2022.103.

25 Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005. doi:10.1017/CBO9780511813603.

26 Yuval Peres, Kunal Talwar, and Udi Wieder. The $(1+\beta)$-choice process and weighted balls-into-bins. In *Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1613–1619, 2010.

27 Martin Raab and Angelika Steger. "Balls into bins" - A simple and tight analysis. In *Randomization and Approximation Techniques in Computer Science, 2nd International Workshop (RANDOM)*, volume 1518 of *Lecture Notes in Computer Science*, pages 159–170, 1998. doi:10.1007/3-540-49543-6_13.

28 Sebastien Roch. *Modern Discrete Probability: An Essential Toolkit*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2024. doi:10.1017/9781009305129.

29 Jan van den Brand, Yu Gao, Arun Jambulapati, Yin Tat Lee, Yang P. Liu, Richard Peng, and Aaron Sidford. Faster maxflow via improved dynamic spectral vertex sparsifiers. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, pages 543–556, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3519935.3520068.

30 Berthold Vöcking. How asymmetry helps load balancing. *Journal of the ACM (JACM)*, 50(4):568–589, 2003. doi:10.1145/792538.792546.

31 Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. Feature hashing for large scale multitask learning. In *Proceedings of the 26th annual international conference on machine learning (ICML)*, pages 1113–1120, 2009.

32 Udi Wieder. Balanced allocations with heterogenous bins. In *Proceedings of the 19th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 188–193, 2007. doi:10.1145/1248377.1248407.

A Uniform two choices game

In this section, we prove Theorem 3.5, which states that if the algorithm uses the two choices allocation scheme, then after $n - n/2^\ell$ rounds of adaptive game, the recourse is $O(\ell n)$ and the maximum load is $O(2^\ell + \ell \log\log n)$.

The proof is similar to that of Theorem 3.1. It uses the following non-adaptive games that use the two choices allocation scheme. In the j-th non-adaptive game, we begin with $n/2^{j-1}$ empty bins. Then, we iteratively draw a pair of indices (i, i') uniformly over $[n/2^{j-1}]^2$, and place a ball in the least loaded among bins i, i'. The game proceeds until every subset $S \subset [n/2^{j-1}]$ of size $|S| = n/2^j$ contains the two indices of at least n pairs. We denote by Y_j the number of inserted balls, and denote by B_j the maximum load of a bin at the end of the game.

▶ **Lemma A.1.** *Let $j \in [\log n]$. Fix an adaptive adversary, and denote the recourse it obtains against two choices allocations in the j-th phase by X_j, and its maximum load in the j-th phase by A_j. Then, $Y_j \succeq X_j$ and $B_j \succeq A_j$, where Y_j and B_j are the number of inserted balls and the maximum load in the j-th non-adaptive game defined above.*

Proof. The proof is similar to the single choice (uniform) game of Lemma 3.3. We first analyze the recourse. We create a coupling between Y_j and X_j, by simulating the non-adaptive game during the $n/2^j$ rounds of the adaptive game. Abusing notation slightly, we continue to use X_j and Y_j to denote these random variables within the coupled setting.

For the non-adaptive game, let L_i^{na} denote the load of the i-th bin for $i \in [n/2^{j-1}]$. In the adaptive game, denote by $\texttt{Alive}_j \subseteq [n]$ the set of living bins right before the j-th phase starts. In particular, $|\texttt{Alive}_j| = n/2^{j-1}$, and therefore there is a one-to-one mapping between $\texttt{Alive}_j$ and $[n/2^{j-1}]$. For simplicity of presentation, we omit this mapping and simply treat every $i \in \texttt{Alive}_j$ as a number in $[n/2^{j-1}]$. Let $\texttt{Alive} \subseteq \texttt{Alive}_j$ denote the set of living bins during phase j; for each $i \in \texttt{Alive}$, let L_i^a be the load of the i-th bin incurred by balls that move in the j-th phase. We maintain the following invariant.

- **Invariant.** For every $i \in \texttt{Alive}$, we have $L_i^{na} \geq L_i^a$.

Initially, in the non-adaptive game there are no balls. No balls moved yet in the adaptive game. Clearly, the invariant holds.

Now, consider a round of the adaptive game. The adaptive adversary picks a bin to delete, say containing b balls. Suppose these b balls are thrown one at a time in the adaptive game, and let us simulate throwing one ball using the non-adaptive game. In the non-adaptive game, to throw a ball, we independently draw two indices $i, i' \in [n/2^{j-1}]$, and allocate the ball to the bin with smaller load (i.e., bin i if $L_i^{na} \leq L_{i'}^{na}$, and bin i' otherwise). If $(i, i') \notin \texttt{Alive}^2$, i.e., bin i or i' were deleted in the adaptive game, we repeatedly throw another new ball to the $n/2^{j-1}$ bins, until we get two indices in $(i, i') \in \texttt{Alive}$. The process of repeatedly throwing new balls may increase the load over $\texttt{Alive}$ in the non-adaptive game, but this doesn't damage the invariant. When obtaining two indices $i, i' \in \texttt{Alive}$, let us assume without loss of generality that $L_i^{na} \geq L_{i'}^{na}$, and so we allocate the ball in bin i' in the non-adaptive game. If $L_i^a \geq L_{i'}^a$ as well, we allocate the ball in bin i' also in the adaptive game, and the invariant holds. Otherwise, $L_i^a < L_{i'}^a$, so we must allocate the ball in bin i in the adaptive game. Before allocating the ball, we have by the invariant, $L_i^a < L_{i'}^a \leq L_{i'}^{na} \leq L_i^{na}$, and since the loads are integers, $L_i^a \leq L_i^{na} - 1$. Thus, the invariant still holds after adding this one ball to bin i in the adaptive game (which increases L_i^a by 1). This process is repeated for all the b balls, and then we proceed to the next round of the adaptive game.

Crucially, we claim that the above is a simulation of the adaptive game, and it may end only after the adaptive adversary deletes $n/2^j$ bins. In the non-adaptive game, the distribution of a pair of indices i, i', given that both land in $\texttt{Alive}$, is uniform over $\texttt{Alive}^2$. Thus, the above indeed simulates the adaptive game.

Observe that right before the adaptive adversary deletes the $\frac{n}{2^j}$-th bin, we have that $|\texttt{Alive}| = \frac{n}{2^j} + 1$. Since $\sum_{i \in \texttt{Alive}} L_i^a \leq n$, there must exists a subset S of $\texttt{Alive}$ of size $\frac{n}{2^j}$, whose bins in the adaptive game contain strictly less than n balls. Hence, in the adaptive game, there were strictly less than n pairs of indices both in S. Therefore, by the construction, since $S \subset \texttt{Alive}$, there were strictly less than n pairs of indices in S also in the non-adaptive game, and thus the non-adaptive game didn't end. Hence, the number Y_j of balls in the non-adaptive game is at least the recourse X_j of the adaptive game in the j-th phase, i.e., $Y_j \geq X_j$.

Over the same probability space, we immediately obtain a similar result for the maximum load. The maximum load B_j in the non-adaptive game is larger than the maximum load in $\texttt{Alive}$, which by the invariant, is larger than the maximum load incurred by balls that move in the j-th phase of the adaptive game, and thus $B_j \geq A_j$. The proof is concluded by Lemma 1.7. ◀

▶ **Lemma A.2.** *Consider the j-th two choices non-adaptive game and let Y_j be the number of balls it throws. Then, $Y_j \leq 24n$ with probability at least $1 - e^{-n}$.*

Proof. The proof is similar to the proof of Lemma 3.4. Consider throwing $24n$ pairs of indices independently and uniformly over $[n/2^{j-1}]^2$. We show that with high probability, all subsets of $n/2^j$ bins contain at least n pairs, and the claim follows.

Let a set $S \subset [n/2^{j-1}]$ of size $|S| = \frac{n}{2^j}$. For every $i \in [24n]$, denote by I_i an indicator that the i-th pair is in S^2. Thus, the number of pairs in S equals $\sum_{i=1}^{24n} I_i$. Clearly, $I_i = 1$ w.p. $\frac{1}{4}$ and 0 otherwise, and all these indicators are independent. Thus, by Hoeffding's inequality,

$$\Pr\Big(\sum_{i=1}^{24n} I_i < n\Big) \leq \Pr\Big(|\sum_{i=1}^{24n} I_i - 6n| > 5n\Big) \leq 2\exp\Big(-\frac{2(5n)^2}{24n}\Big) \leq 2\exp(-2n).$$

By a union bound, the probability that there exists a set $S \subset [n/2^{j-1}]$ of size $|S| = \frac{n}{2^j}$ containing less than n balls is at most

$$\binom{n/2^{j-1}}{n/2^j} \cdot 2\exp(-2n) \leq 2^n \cdot 2\exp(-2n) \leq \exp(-n),$$

which concludes the proof of the lemma. ◀

Proof of Theorem 3.5. By combining Lemma A.1 and Lemma A.2, we have that for each $j \in [\ell]$, with probability at least $1 - e^{-n}$, the recourse is at most $24n$. By a union bound, with probability at least $1 - e^{-n/2}$, the total recourse is $O(\ell n)$. In standard two choices allocations with $10n$ balls and $n/2^{j-1}$ bins, by [11], with probability $1 - \frac{1}{n^{11}}$, the maximum load is $B_j = O(2^j + \log\log n)$.

Thus, by a union bound, with probability $1 - n^{-10}$, the maximum load in the non-adaptive game is

$$\leq \sum_{j=1}^{\ell} A_j \leq \sum_{j=1}^{\ell} B_j \leq O(2^\ell + \ell \log\log n),$$

and the proof follows. ◀

B Proof of Lemma 4.7

To prove Lemma 4.7, we apply McDiarmid's inequality.

▶ **Lemma B.1** (McDiarmid's inequality). *Suppose a function $f : \mathbb{R}^n \to \mathbb{R}$ satisfies for all $i \in [n]$, and $x_1, \ldots, x_n \in \mathbb{R}$,*

$$\sup_{x_i' \in \mathbb{R}} \left| f(x_1, \ldots, x_i, \ldots, x_n) - f(x_1, \ldots, x_i', \ldots, x_n) \right| \leq c.$$

Let $X_1, \ldots, X_n \in \mathbb{R}$ be independent random variables. For all $t > 0$,

$$\Pr\left(|f(X_1, \ldots, X_n) - \mathbb{E}f(X_1, \ldots, X_n)| \geq t\right) \leq 2\exp\Big(-\frac{2t^2}{nc^2}\Big).$$

Recall that in Lemma 4.7, $10n$ balls are thrown to n bins, the bins loads are denoted by $L_1, L_2, \ldots, L_n$, and the claim is that with high probability, $\sum_{i=1}^n L_i \log L_i = O(n)$. Denote $F = \sum_{i=1}^n L_i \ln L_i$, which is $O(1)$ factor from the desired sum. We view F as a function of the locations of the $10n$ balls. Consider the effect of changing the location of one ball, say, by changing its location from bin i to bin j. Then F changes by

$$\begin{aligned}
&\left|(L_i - 1)\ln(L_i - 1) + (L_j + 1)\ln(L_j + 1) - L_i \ln L_i - L_j \ln L_j\right| \\
&= |L_i \ln(\tfrac{L_i - 1}{L_i}) + L_j \ln(\tfrac{L_j + 1}{L_j}) - \ln(L_i - 1) + \ln(L_j + 1)| \\
&\leq 2\ln(10n) + |L_i \ln(\tfrac{L_i - 1}{L_i})| + |L_j \ln(\tfrac{L_j + 1}{L_j})| \\
&\leq 2\ln(10n) + 2,
\end{aligned}$$

where the last inequality is since $\ln(1 + x) \leq x$ for $x > -1$.

▷ Claim B.2. We have $\mathbb{E}F \leq n \cdot 10 \ln 11 < 25n$.

Proof of Lemma 4.7. By McDiarmid's inequality,

$$\begin{aligned}\Pr(F > 100n) &\le \Pr(F - \mathbb{E}(F) > 75n)\\ &\le 2\exp\Big(-\frac{2(75n)^2}{10n(2\ln(10n+2)+2)^2}\Big) \le e^{-\sqrt{n}}.\end{aligned}$$

◀

Proof of Claim B.2. By linearity of expectation, $\mathbb{E}F = n \cdot \mathbb{E}(L_1 \ln L_1)$. Observe that $L_1 \sim Bin(10n, \frac{1}{n})$, i.e., it is a binomial random variable with $10n$ experiments and success probability $\frac{1}{n}$. Therefore,

$$\begin{aligned}\mathbb{E}L_1 \ln L_1 &= \sum_{k=1}^{10n}\binom{10n}{k}\Big(1-\frac{1}{n}\Big)^{10n-k}\frac{1}{n^k}\cdot k\ln k\\ &= \sum_{k=1}^{10n}\frac{10n}{k}\binom{10n-1}{k-1}\Big(1-\frac{1}{n}\Big)^{10n-k}\frac{1}{n^k}\cdot k\ln k && \text{since } k\binom{m}{k} = m\binom{m-1}{k-1}\\ &= 10\sum_{k=1}^{10n}\binom{10n-1}{k-1}\Big(1-\frac{1}{n}\Big)^{10n-k}\frac{1}{n^{k-1}}\cdot \ln k\\ &= 10\sum_{k=0}^{10n-1}\binom{10n-1}{k}\Big(1-\frac{1}{n}\Big)^{10n-1-k}\frac{1}{n^k}\cdot \ln(k+1) && \text{change of index}\\ &= 10\cdot\mathbb{E}\ln(X+1) && \text{where } X\sim Bin(10n-1,\tfrac{1}{n})\\ &\le 10\ln(\mathbb{E}X+1) && \text{by Jensen's inequality}\\ &\le 10\ln 11 < 25.\end{aligned}$$

◁

Arboricity Matters in Triangle Counting with Random Edges

Arijit Bishnu
Indian Statistical Institute, Kolkata, India

Debarshi Chanda
Indian Statistical Institute, Kolkata, India

Gopinath Mishra
Institute of Mathematical Sciences, Chennai, India

Abstract

Given a simple, unweighted, undirected graph $G = (V, E)$ with $|V| = n$ and $|E| = m$, and parameters $0 < \varepsilon, \delta < 1$, along with `Degree`, `Neighbour`, `Pair` and `RandomEdge` query access to G, we provide a query-based randomized algorithm to generate an estimate $\widehat{T}$ of the number of triangles T in G, such that $\widehat{T} \in [(1-\varepsilon)T, (1+\varepsilon)T]$ with probability at least $1-\delta$. The query complexity of our algorithm is $\widetilde{O}\left(m\alpha \log(1/\delta)/\varepsilon^3 T\right)$, where α is the arboricity of G. Our work can be seen as a natural progression to the line of recent works [Eden et al., SIAM J Comp., 2017; Assadi et al., ITCS 2019; Eden et al., SODA 2020] that considered subgraph or triangle counting with or without the use of `RandomEdge` query. Of these works, Eden et al. [SODA 2020] considers the role of arboricity. Our work is the first to consider how `RandomEdge` query can leverage the structural property of arboricity. Furthermore, continuing in the line of work of Assadi et al. [APPROX/RANDOM 2022], we also provide a lower bound of $\widetilde{\Omega}\left(m\alpha \log(1/\delta)/\varepsilon^2 T\right)$ that matches the upper bound exactly on arboricity, δ, and almost on ε.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases Triangle counting, Sublinear algorithms, Subgraph counting

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.55

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2502.15379`

1 Introduction

Counting the number of triangles present in graphs is a classical problem [20, 5, 53, 4, 10]. Researchers in the streaming (sub-linear space) community have studied this problem for the last two decades; see [10, 11, 35, 2, 17, 6, 36, 18, 22, 55, 41] for a sample of such works. The problem has also been studied by researchers in the property testing (sub-linear time) community in the last decade [26, 48, 39, 11, 13, 28, 35]. Here, graphs can be accessed only through certain queries. On the other hand, starting with the work of [20], the graph parameter called arboricity has also found its use in triangle counting [28]. Our work shows the use of arboricity in triangle counting where the graph can be accessed through random edge queries along with other standard queries like `Degree`, `Neighbour` and `Pair`.

Counting triangles has been a well-motivated problem with applications across various domains like optimizing query size in database join problems [10, 9, 7], computing clustering coefficients, transitivity ratio [1, 46, 60, 44], studying structures in web graphs [25, 23] etc. For a more thorough overview, see the surveys [3, 57]. The parametrization based on arboricity is also of practical interest due to the occurrence of low-arboricity graphs in various real-world scenarios [24, 42, 50, 31, 12, 23, 54].


© Arijit Bishnu, Debarshi Chanda, and Gopinath Mishra;
licensed under Creative Commons License CC-BY 4.0

Formally, we are given a simple, unweighted, undirected graph $G = (V, E)$ with $|V| = n$, and $|E| = m$. The neighbours of v is denoted $\mathsf{N}(v) = \{u | (u, v) \in E\}$ and the degree of the vertex v is denoted $\deg_v = |\mathsf{N}(v)|$. We are given the following queries to access the graph:

- `Degree`(v): Given a vertex v, return $\deg_v$.
- `Neighbour`(v, i): Given a vertex $v \in V$ and $i \in [n]$, return the i-th vertex $u \in \mathsf{N}(v)$ if it exists; otherwise, we get $\perp$.
- `Pair`(u, v): Given two vertices $u, v \in V$, return 1 if $(u, v) \in E$, and 0, otherwise.
- `RandomEdge`: Return an edge $e \in E$ uniformly at random.

We will call the `Degree`, `Neighbour` and `Pair` queries taken together to be *local queries* [7]. Now, we formally state our problem:

Triangle Estimation:
Given ε, δ with $0 < \varepsilon, \delta < 1$, obtain an estimate $\widehat{T}$ of the number of triangles T in the graph G, such that $\widehat{T} \in [(1-\varepsilon)T, (1+\varepsilon)T]$ with probability at least $1 - \delta$.

From now on, we refer $\widehat{T}$ to be an (ε, δ) estimate of T. One of the graph parameters that has been relevant to quantify the complexity of triangle counting algorithms within the context of RAM model [20], streaming model [13], distributed model [34, 56, 30, 45] and query model [29], is the arboricity of the graph G, denoted α. The arboricity of a graph can be seen as a measure of the density of the graph.

▶ **Definition 1** (Arboricity(α)). *The arboricity of a graph $G = (V, E)$, denoted α, is the minimum number of spanning forests that E can be partitioned into.*

Recent works have established various ways to construct lower bounds incorporating the impact of ε and δ both in graph property testing [8] and other problems [58]. Our lower bounds will also involve ε and δ. To illustrate the dependence of ε, δ, and polylog terms on the query complexity of our algorithm, we will use two notations at times in place of $O(\bullet)$ – $\widetilde{O}(\bullet)$ hides polylog factors only but shows dependence on ε and δ, whereas $\widetilde{O}_{\varepsilon,\delta}(\bullet)$ hides both ε, δ and polylog terms.

1.1 Our results

In this section, we present the main results of this work.

Upper bound

We show that using random edge queries in addition to local queries, we can obtain an (ε, δ) estimate of T where the query complexity is parametrized by arboricity apart from the usual parameters of edge, vertex, the number of triangles, ε, and δ. The work of [20] established the role of arboricity in the problem of triangle counting giving an $O(m\alpha)$ time algorithm to exactly count the number of triangles T in the RAM model. In recent works in the property testing model, [26] gave an algorithm that uses local queries and requires $\widetilde{O}_{\varepsilon,\delta}\left(\frac{n}{T^{1/3}} + \min\left\{m, \frac{m^{3/2}}{T}\right\}\right)$ queries to obtain an (ε, δ) estimate $\widehat{T}$ of T. Using `RandomEdge` in addition to the local queries, [7] proposed an algorithm that uses $\widetilde{O}\left(\frac{m^{3/2}\log(1/\delta)}{\varepsilon^2 T}\right)$ queries. Following this, [28] was introduces the parameter of arboricity to triangle counting in the query based setup. They proposed an algorithm that uses $\widetilde{O}\left(\frac{n}{T^{1/3}} + \frac{m\alpha\log^3(1/\delta)}{\varepsilon^5 T}\right)$ local queries to obtain an (ε, δ) estimate $\widehat{T}$ of T. Our work follows in the line of the works of [26, 7, 28], where we use `RandomEdge` in addition to *local* queries to obtain an algorithm that uses $\widetilde{O}\left(\frac{m\alpha\log(1/\delta)}{\varepsilon^3 T}\right)$ queries (see the last row in Table 1). Formally,

▶ **Theorem 2** (Upper bound). *Given a simple, unweighted and undirected graph $G = (V, E)$ with $|V| = n$, $|E| = m$ and arboricity α, and access to the graph via* `Degree`, `Neighbour`, `Pair` *and* `RandomEdge` *queries, an (ε, δ) estimate $\widehat{T}$ of T can be obtained using $\widetilde{O}\left(\frac{m\alpha \log(1/\delta)}{\varepsilon^3 T}\right)$ queries.*

A constant-pass $\widetilde{O}\left(\frac{m\kappa}{\varepsilon^4 T}\right)$ space streaming algorithm was proposed in [13] for obtaining an (ε, δ)-estimate of the number of triangles in a graph, where κ is the degeneracy of the graph. Our algorithm can be directly extended to a constant pass streaming algorithm using $\widetilde{O}\left(\frac{m\alpha}{\varepsilon^3 T}\right)$ space. By the fact that $\alpha \leq \kappa \leq 2\alpha$, our algorithm's space complexity is as good as the algorithm of [13] in terms of the graph parameters, and asymptotically better in terms of the approximation factor ε.

Lower bound

We also establish a lower bound involving the arboricity and the parameters ε and δ. Reductions from communication complexity is a well studied method to establish lower bound results in terms of query complexity [15, 29, 32]. Using only `Degree` and `Neighbour` queries, triangle counting requires $\Omega(m)$ queries [33]. Including `Pair` with the previous two queries, [26] established a lower bound of $\Omega\left(\frac{n}{T^{1/3}} + \frac{m^{3/2}}{T}\right)$ queries which matches their upper bound up to $poly\left(\frac{\log(n/\delta)}{\varepsilon}\right)$ factors. Later, a simpler proof was proposed for the same bound through reduction from communication complexity problems [29]. This was subsequently extended to incorporate arboricity with a bound of $\Omega\left(\frac{n}{T^{1/3}} + \frac{m\alpha}{T}\right)$ queries [28] which again matched their proposed upper bound up to $poly\left(\frac{\log(n/\delta)}{\varepsilon}\right)$ factors. If `RandomEdge` query is allowed along with all three local queries, the above lower bounds of $\Omega\left(\frac{n}{T^{1/3}} + \frac{m^{3/2}}{T}\right)$ and $\Omega\left(\frac{n}{T^{1/3}} + \frac{m\alpha}{T}\right)$ become $\Omega\left(m^{3/2}/T\right)$ [29] and $\Omega(m\alpha/T)$ [28], respectively. [1] As we have mentioned, for general graphs, there is an algorithm that makes $\widetilde{O}\left(\frac{m^{3/2}\log(1/\delta)}{\varepsilon^2 T}\right)$ queries, where each query is either a local query or a `RandomEdge` query [7]. Recently, in [8], it has been shown that the algorithm of [7] is optimal (up to $poly(\log(n))$ factors) in terms of all parameters including ε and δ through a lower bound of $\Omega\left(\frac{m^{3/2}\log(1/\delta)}{\varepsilon^2 T}\right)$ queries for obtaining an (ε, δ) estimate of T using local and `RandomEdge` queries. However, this does not consider the dependency of the query complexity on arboricity of the graph. Like our upper bound, the lower bound of our work also captures the role of arboricity and the parameters ε, and δ when the queries allowed are the local and the `RandomEdge` queries. Moreover, our lower bound combines the parametrization of [8] and [28] in terms of ε, δ, and α, respectively, to establish an improved lower bound.

▶ **Theorem 3** (Lower bound). *Given a simple, unweighted and undirected graph $G = (V, E)$ with $|V| = n$, $|E| = m$ and arboricity α, and access to the graph via* `Degree`, `Neighbour`, `Pair` *and* `RandomEdge` *queries, obtaining an (ε, δ) estimate $\widehat{T}$ of T requires $\Omega\left(\frac{m\alpha \log(1/\delta)}{\varepsilon^2 T}\right)$ queries.*

Table 1 describes the upper and lower bounds across different query accesses and parametrization for the triangle counting problem, with our contributions in this work highlighted with an asterisk $(*)$ mark. Table 1 shows that it is only natural to study the problem of

[1] Though [28] does not mention about $\Omega(m\alpha/T)$ explicitly, it is not difficult to see it from their construction. Also, this lower bound of $\Omega(m\alpha/T)$ matches with our upper bound (stated in Theorem 2) in terms of dependence on m, T, and α.

Table 1 Upper and lower bounds for the Triangle Counting problem. Entries marked with an asterisk (∗) are our results proved in this paper.

	Without `RandomEdge`	**With `RandomEdge`**
Without Arboricity	UB: $\widetilde{O}_{\varepsilon,\delta}\left(\frac{n}{T^{1/3}} + \frac{m^{3/2}}{T}\right)$ [26] LB: $\widetilde{\Omega}\left(\frac{m^{3/2}\log(1/\delta)}{\varepsilon^2 T}\right)$ [8]	UB: $\widetilde{O}\left(\frac{m^{3/2}\log(1/\delta)}{\varepsilon^2 T}\right)$ [7] LB: $\widetilde{\Omega}\left(\frac{m^{3/2}\log(1/\delta)}{\varepsilon^2 T}\right)$ [8]
With Arboricity	UB: $\widetilde{O}\left(\frac{n}{T^{1/3}} + \frac{m\alpha\log^3(1/\delta)}{\varepsilon^5 T}\right)$ [28] (∗) LB: $\Omega\left(\frac{m\alpha\log(1/\delta)}{\varepsilon^2 T}\right)$	(∗) UB: $\widetilde{O}\left(\frac{m\alpha\log(1/\delta)}{\varepsilon^3 T}\right)$ (∗) LB: $\Omega\left(\frac{m\alpha\log(1/\delta)}{\varepsilon^2 T}\right)$

triangle estimation with the use of `RandomEdge` and arboricity as a parameter after the works of [26, 28, 7, 8]. We believe our work also raises some new questions: (a) it opens up the question of quantifying the impact of α to count other subgraphs (e.g., k-cliques) using `RandomEdge` and other local queries; (b) there remains a small gap of $1/\varepsilon$ between our upper and lower bounds that remain to be resolved.

1.2 Related work in other models

The problem of triangle counting is a problem that has been extensively studied across various models of computation. Here, we discuss the works in RAM and streaming models.

RAM model. The work of [20] proposed an $O\left(\frac{m\alpha}{T}\right)$ algorithm in the RAM model that remains the best known till date. Recent works have shown it to be conditionally optimal under the 3SUM [43] and APSP hypotheses [59].

Streaming model. In the data streaming model, a long line of work [51, 52] has culminated in an $\widetilde{O}\left(\frac{m}{T}\left(\Delta_E + \sqrt{\Delta_V}\right)\right)$-space one-pass streaming algorithm for insertion only model [39] that has been shown to be optimal through the combined lower bounds proposed in [16, 39]. Here, Δ_E (Δ_V) denote the maximum number of triangles incident on an edge e (a vertex v) in the graph G. There have been works on triangle counting in multi-pass setup as well as in other streaming models (e.g., turnstile, cash-register etc.) [48, 11, 39, 37, 38, 40, 13].

1.3 Paper organization

Section 2 discusses a broad overview of the techniques we use, and contextualizes it with respect to the relevant prior works. Section 3 sets up the preliminaries for our proofs, including the structural results involving arboricity of a graph. Section 4 discusses the query-based algorithm by breaking the discussion into several parts: a weight function (Section 4.1), an oracle based algorithm (Sections 4.2) and its implementation (Section 4.3) and the final algorithm (Section 4.4). The lower bounds are discussed in Section 5. Some of the proofs are deferred to the Appendix. We have also included the full version of the paper in the Appendix.

2 Technical overview

In this section, we give a broad overview of the techniques used in this work. We denote by T_e the number of triangles incident on the edge e, and the degree of an edge by $\deg_e = \deg_{(u,v)} = \min\{\deg_u, \deg_v\}$.

2.1 Upper bound

Our starting point is to use `RandomEdge` queries to obtain S, a random sample of edges of size s and try to estimate the number of triangles T incident on the edges of S. As the `RandomEdge` query samples edges uniformly at random, the expected number of triangles each sampled edge participates in is $\frac{T}{m}$. Hence, an obvious approach to estimate T would be to count the number of triangles T_e each edge e participates in, and take their average for the sampled edges and scale it appropriately. However, there are two issues with this approach.

Firstly, counting the number of triangles an edge e participates in exactly would require $\Omega(\deg_e)$ *local* queries. Therefore, we have to sample sufficiently many neighbour vertices for each edge e and check if it forms a triangle or not to obtain a good estimate of T_e.

Secondly, even though the expected number of triangles incident on each edge is $\frac{T}{m}$, individual edges e can participate in a large number of triangles, i.e., T_e can be $\Omega(T)$. This makes the variance of the estimate too high, and requires s to be large to obtain a good estimate. To work around this issue, we fix a threshold τ based on the arboricity α of the graph, that has a connection to the number of triangles in the graph. We call edges that participates in more than τ triangles *τ-heavy edges*. Correspondingly, any edge that participates in less than τ triangles is called *τ-light edges*. For now, assume we are given access to an oracle `Heavy` to decide whether an edge is light or heavy. We call the triangles consisting exclusively of heavy edges as *heavy triangles*, and the remaining triangles as *light triangles*. The threshold τ is set at an appropriate multiple of α so that heavy triangles constitute only ε fraction of the triangles. Thus, we can forget about the *τ-heavy edges* and focus only on counting the light triangles. This brings us to the issue of sampling light triangles uniformly at random. We define a weight function on the edges that assigns each light triangle to one of its constituent light edges. The weight function is defined to be the number of light triangles assigned to the edge, with each heavy edge being assigned a value of 0.

Now the algorithm's objective is to estimate this weight function, which cannot be accessed or sampled directly. Observe that there can be multiple assignments of the light triangles to the edges, and consequently multiple weight functions. It suffices for the algorithm to estimate any one of them. In that regard, for all the light edges in our sample S, we maintain a partial non-conflicting assignment. Such an assignment can be extended to a valid assignment, and consequently, a valid weight function. Based on this assignment, we obtain an estimate of the corresponding weight function, which, given an appropriately fixed threshold $\tau = \Theta\left(\frac{\alpha}{\varepsilon}\right)$, and setting s to $\widetilde{O}\left(\frac{m\alpha}{\varepsilon^3 T}\log\frac{1}{\delta}\right)$ will give an (ε, δ) estimate of T.

To design the oracle `Heavy` for a given edge e, we estimate the number of triangles each edge participates in. We can estimate it by sampling each triangle independently through the corresponding neighbouring vertex of e. The probability of obtaining a triangle for an edge e through a random neighbour is $\frac{T_e}{\deg_e}$. Hence, for the heavy edges where $T_e \geq \tau$, T_e can be estimated using sufficiently small number of queries. For the light edges where $T_e \leq \tau$, the probability of obtaining a triangle through a random neighbour can be arbitrarily low. However, note that the lower probability of obtaining triangles allows for high approximation error. We then design a bucketing trick to exploit this trade-off to implement an efficient algorithm to decide whether an edge is heavy or light. For full details, see Section 4.

2.2 Lower bound

Several lower bounds for triangle counting problems based on reductions from the *disjointness* problem in communication complexity [29, 7, 28] are often close to the best possible upper bounds. It is not clear whether this approach can give tight bounds involving ε and δ. Recently, Assadi and Nguyen [8] showed that any algorithm to find an (ε, δ)-estimate of the number of triangles T must make at least $\Omega\left(\frac{m^{3/2}\log(1/\delta)}{\varepsilon^2 T}\right)$ queries, where each query is either a local query or a `RandomEdge` query.[2] To prove the lower bound, they introduced a new query problem called the *Popcount Thresholding Problem* (`PTP`), and gave reductions from this problem to triangle estimation, instead of using disjointness. We also show a reduction from `PTP`. In particular, we show that for any value of arboricity α, we can design a graph G with arboricity α such that finding an (ε, δ)-estimate in it using $\Omega\left(\frac{m\alpha\log(1/\delta)}{\varepsilon^2 T}\right)$ queries would violate the lower bound of `PTP`.

In the $\texttt{PTP}_{M,k,\gamma}$ problem, we are given query access to a binary string $x \in \{0,1\}^M$ and must decide whether it was generated from one of two similar distributions. In the first distribution, denoted $\mathcal{D}_0$, each bit of the string is independently set to 1 with probability $(1-2\gamma)k/M$; in the second distribution, denoted $\mathcal{D}_1$, each bit is independently set to 1 with probability $(1+2\gamma)k/M$. The objective is to distinguish between these two cases by adaptively querying bits of x. For suitable choice of k,γ, and δ, with probability at least $1-\delta$, a string from $\mathcal{D}_0$ will have fewer than $(1-\gamma)k$ 1s and a string from $\mathcal{D}_1$ will have more than $(1+\gamma)k$ 1s. Hence, to distinguish whether x is generated from $\mathcal{D}_0$ or $\mathcal{D}_1$, it suffices to estimate the number of 1s present in x, i.e. $\|x\|_1$ upto multiplicative approximation factor γ. Assadi et al. [8] showed that (under suitable choice of parameters) $\Omega\left(\frac{M\log(1/\delta)}{\gamma^2 k}\right)$ queries to x are needed to distinguish between the case whether $x \sim \mathcal{D}_0$ and $x \sim \mathcal{D}_1$ with probability $1-\delta$.

We now describe the hard graph instance $G_x(V_x, E_x)$ used in the reduction. The core structure of V_x consists of three vertex sets: A, B, and R, where $|A| = |B| = \Theta(m/\alpha)$ and $|R| = \Theta(\alpha)$. Every vertex in $A \cup B$ is connected to all vertices in R, forming a bipartite clique between $A \cup B$ and R. The subgraph induced by $A \cup B$ contains $\Theta(m)$ edges, with each vertex having degree at most $O(\alpha)$. We associate each index $i \in [M]$ from the $\texttt{PTP}_{M,k,\gamma}$ instance with a potential edge between A and B, and include the edge if and only if $x_i = 1$. Thus, the number of edges between A and B equals $\|x\|_1$, and the total number of triangles in the graph G_x is $\|x\|_1 \cdot |R|$. By choosing parameters such that $M = \Theta(m)$, $k = \Theta(T/\alpha)$, and $\gamma = \Theta(\varepsilon)$, distinguishing whether x is drawn from $\mathcal{D}_0$ or $\mathcal{D}_1$ becomes equivalent to distinguishing whether the number of triangles in the graph is at most $(1-\varepsilon)T$ or at least $(1+\varepsilon)T$. To complete the reduction, we show that each local query and `RandomEdge` query to G_x can be simulated using only a constant number of queries to x. To enable this, we extend the construction by introducing two additional disjoint vertex sets A' and B', resulting in five disjoint sets: A, B, R, A', and B'. All triangles in the graph are confined to the sets A, B, and R, while the sets A' and B' are added to ensure that the degree of each vertex in V_x remains identical regardless of whether x is drawn from $\mathcal{D}_0$ or $\mathcal{D}_1$. With this construction, every allowable query to the graph can be simulated with $O(1)$ queries to x. For full details, see Section 5.

[2] Note that this lower bound matches exactly with the corresponding upper bound in [7].

2.3 Contextualizing with relevant prior work

Our work follows as a natural progression to the works of [26, 7, 28] in the property testing framework. So, let us review the algorithmic techniques at play in them.

We start by describing a simple randomized algorithm due to Assadi et al. [7] for estimating the number of triangles in a graph when we have access to queries like `Degree`, `Neighbour`, `Pair`, and `RandomEdge`. First, pick a random edge $\{u, v\}$ and let $\deg_u \leq \deg_v$. If $\deg_u \leq \tau_d$, pick a random neighbour w of u and check if $\{u, v, w\}$ forms a triangle in the correct order, i.e., in the increasing degree sequence. The probability of sampling a triangle in this case is $\frac{1}{2m} \cdot \frac{1}{\deg_u} \cdot \frac{\deg_u}{\tau_d} = \frac{1}{2m\tau_d}$. If $\deg_u > \tau_d$, pick a vertex w at random with probability proportional to its degree. This is possible because `RandomEdge` allows sampling vertices proportional to their degrees. The probability of sampling a triangle in this case is $\frac{1}{2m} \cdot \frac{\deg_w}{2m} \geq \frac{\tau_d}{4m^2}$. By setting $\tau_d = \sqrt{2m}$, we can ensure that the algorithm works effectively. This process finds a triangle with probability $\Omega(T/m^{3/2})$. Finally, by repeating this process $\widetilde{O}(m^{3/2} \log(1/\delta)/\varepsilon^2 T)$ times, we can estimate the total number of triangles.

The setting $\tau_d = \Theta(\sqrt{m})$ seems to be the optimal choice for the general case. We do not see a way to generalize the above idea when parametrized by arboricity. In bounded arboricity graphs, the structure and degree distribution of the graph may not align with this approach. So, a different strategy would be required for graphs with bounded arboricity to achieve the desired bound of $\widetilde{O}(m\alpha \log(1/\delta)/\varepsilon^3 T)$.

Our approach is more closely aligned with the framework of triangle counting developed by Eden et al. [26] with access to local queries only. The algorithm by Eden et al. [26] starts by picking a small set of vertices at random and for each sampled vertex, the algorithm tries to estimate the number of triangles it participates in. Some vertices appear in many triangles and are called heavy, while others are called light. To keep the estimate as desired, the algorithm limits the contribution of heavy vertices and focuses mainly on light ones. It gives more weight to triangles that have more light vertices. Since the number of triangles a vertex participates in cannot be obtained directly, the algorithm estimates this by sampling edges incident on the sampled vertices and checking if they close triangles with neighbouring vertices of those edges. The above framework has also been subsequently extended for clique counting in general graphs [28] and that in bounded arboricity graphs [28].

In summary, the idea of dividing the triangles in terms of their constituent edges or vertices into heavy and light, and dealing with them separately has been a long-standing technique for subgraph counting in sublinear models [48, 26, 11, 27, 38, 28, 13]. The algorithmic techniques vary on how to set and leverage this thresholding behaviour for the specific problems and models. Furthermore, allowing `RandomEdge` queries has yielded simpler algorithms for subgraph counting problems [7]. The usage of `RandomEdge` enables us to design a simple algorithm that uses the structural parameter of arboricity.

The lower bounds for the triangle counting problem in the query model have come from indistinguishability arguments [26] or reduction from communication problems, particularly *set-disjointness* [29, 7, 28] with [28] establishing arboricity-sensitive lower bounds. On the other hand, several recent works [14, 8] have established lower bounds for property testing problems through reductions to query problems that are sensitive to ε and δ. In this work, we extend the ε, δ-sensitive lower bound of [8] to the triangle counting problem for any value of arboricity α.

3 Preliminaries

3.1 Notations

The set of positive integers $\{1, 2, \ldots, x\}$ is denoted $[x]$. The set of triangles in G is denoted Δ, and individual triangles are denoted $\mathtt{t}$. (v, e) denotes a triangle formed by the vertices v and the endpoints of the edge e. $\text{Unif}(S)$ denotes an element of S is chosen uniformly at random.

3.2 Arboricity and its properties

As arboricity plays a crucial role in our work, we put together all the structural results that involve arboricity here. Let us restate the definition once more.

▶ **Definition 4** (Arboricity(α)). *The arboricity of a graph $G = (V, E)$, denoted by α_G, is the minimum number of spanning forests that E can be partitioned into.*

The arboricity of a graph can be seen as a measure of the density of the graph. α_G can be at least $\lceil m/(n-1) \rceil$. Also, $\alpha_G \geq \alpha_H$ where H is any subgraph of G. We will write α instead of α_G when the underlying graph is understood. We introduce the following result due to [20] on the sum of edge degrees over all edges in the graph.

▶ **Lemma 5** (Sum of deg_e [20]). *Given a graph $G = (V, E)$ with arboricity α and $|E| = m$, $\sum_{e \in E} deg_e \leq 2m\alpha$.*

The following result due to [28] builds on the work of [20] to bound the number of triangles based on the number of edges m and arboricity α.

▶ **Lemma 6** (Triangle upper bound [28]). *Given a graph $G = (V, E)$ with arboricity α and $|E| = m$, the graph G has at most $m\alpha$ triangles.*

Note that this upper bound is also tight, i.e., there exists graphs that contain m edges and $\Omega(m\alpha)$ triangles. Additionally, arboricity α can be at most $O(\sqrt{m})$ [28]. Thus, all our results can be reformulated by plugging in this upper bound.

3.3 Chernoff bounds

We will be using the following variation of the Chernoff bound that bounds the deviation of the sum of independent Poisson trials [49].

▶ **Lemma 7** (Multiplicative Chernoff bound). *Given i.i.d. random variables $X_1, X_2, ..., X_t$ where $\Pr[X_i = 1] = p$ and $\Pr[X_i = 0] = (1 - p)$, define $X = \sum_{i \in [t]} X_i$. Then, we have:*

$$\Pr[X \leq (1-\varepsilon)\,\mathbb{E}[X]] \leq \exp\left(-\frac{\mathbb{E}[X]\,\varepsilon^2}{3}\right) \qquad 0 \leq \varepsilon < 1$$

$$\Pr[X \geq (1+\varepsilon)\,\mathbb{E}[X]] \leq \exp\left(-\frac{\varepsilon^2\,\mathbb{E}[X]}{2+\varepsilon}\right) \qquad 0 \leq \varepsilon$$

4 Algorithm

This section describes the algorithm and its related concepts. Section 4.1 formally defines the weight function for edges and associated structural results. Section 4.2 describes the algorithm assuming access to an idealized oracle that can decide if an edge is heavy or light

(based on whether the edge has many or few incident triangles). Section 4.3 describes how to actually implement this oracle within the problem setup. Section 4.4 puts everything together to develop the final algorithm.

4.1 Weight function

In this section, we formalize the ideas of heavy and light edges and weight function for the edges.

Heavy and light edges and triangles

First, we define heavy and light edges and correspondingly, heavy and light triangles.

▶ **Definition 8** (τ-heavy and τ-light edges). *An edge $e \in E$ is defined to be a τ-heavy (resp. τ-light) edge if it participates in more than τ (resp. $\leq \tau$) triangles.*

▶ **Definition 9** (τ-heavy and τ-light triangles). *A triangle $\mathsf{t} \in \Delta$ is called a τ-heavy triangle if all its three edges are τ-heavy edges. A triangle that is not τ-heavy is a τ-light triangle.*

We denote by T_L^τ and T_H^τ the number of τ-light and τ-heavy triangles in the graph G, respectively. The following lemma bounds the number of τ-heavy triangles in a graph using the parameter of arboricity.

▶ **Lemma 10** (Upper bound on T_H^τ). *Given a graph $G = (V, E)$ with T triangles, the number of τ-heavy triangles is at most $\frac{3T\alpha}{\tau}$.*

Proof. Note that the graph $G = (V, E)$ has T triangles containing at most $3T$ edges. By Definition 9, τ-heavy triangles have all three of their edges to be τ-heavy edges, τ or more triangles incident on each of them. Hence, the number of τ-heavy edges in G is at most $\frac{3T}{\tau}$.

Now consider the subgraph $H = (V_H, E_H)$ of G induced by the τ-heavy edges in G. We know, $E_H \leq \frac{3T}{\tau}$. Also, $\alpha_H \leq \alpha_G = \alpha$ (see Section 3.2). Hence, by Lemma 6, we know that H contains at most $\frac{3T\alpha}{\tau}$ triangles. ◀

The following corollary follows from Lemma 10 and the fact that $T = T_L^\tau + T_H^\tau$.

▶ **Corollary 11** (Lower bound on T_L^τ). *Given a graph $G = (V, E)$, there are at least $(1 - \frac{3\alpha}{\tau})T$ τ-light triangles.*

Weight function for the edges

Now, we define a weight function for the edges. As a triangle is incident to multiple edges, the objective of a weight function is to charge each of the light triangles to exactly one of its participating light edges. To ensure this, we do not charge any light triangle to any heavy edge that it may be incident on. Each light edge e can be charged by at most T_e triangles, i.e., all the triangles the edge e participates in. To avoid over-counting, we ensure that the sum of the weight function over all edges is equal to the number of light triangles, T_L^τ. (t, e) denotes that the triangle t is charged to the edge e.

▶ **Definition 12** (Triangle weight function (w)). *We define a function $w : E \to \mathbb{N}$ to be a triangle weight function if it satisfies the following two conditions:*

$$\begin{aligned} &\textit{Condition (1)}: \quad w(e) \quad \leq \quad \begin{cases} T_e & \textit{if } e \textit{ is a light edge} \\ 0 & \textit{if } e \textit{ is a heavy edge} \end{cases} \\ &\textit{Condition (2)}: \quad \sum_{e \in E} w(e) \quad = \quad T_L^\tau \end{aligned}$$

Observe that there are multiple such weight functions (e.g., a triangle with 3 light edges can be assigned to any one of these 3 edges). Henceforth, we denote the set consisting of valid triangle weight functions by $\mathcal{W}$. We now state a property of valid weight functions. Note that this property is true for any $w \in \mathcal{W}$.

▶ **Lemma 13** (Expectation of $w(e)$)**.** *Consider any triangle weight function $w \in \mathcal{W}$. If we select an edge $e \in E$ uniformly at random, then the expected value of $w(e)$ is $\frac{T_L^\tau}{m}$, i.e., $\mathbb{E}_{e\sim \mathrm{Unif}(E)} w(e) = \frac{T_L^\tau}{m}$.*

Proof. Given that the edges have been chosen uniformly at random and the condition of the triangle weight function (Definition 12), we have:

$$\mathop{\mathbb{E}}_{e\sim \mathrm{Unif}(E)} w(e) \;=\; \sum_{e\in \mathrm{Unif}(E)} \frac{1}{m} w(e) \;=\; \frac{1}{m}\sum_{e\in E} w(e) \;=\; \frac{T_L^\tau}{m}. \qquad \blacktriangleleft$$

4.2 Oracle based algorithm

Our algorithm, due to its usage of the *triangle weight function*, requires knowledge of whether an edge is *heavy* or *light*. The problem of deciding whether an edge is heavy is non-trivial as it directly relates to number of triangles the edge participates in. In this section, we assume black-box access to an oracle `HeavyOracle` that helps us determine if an edge is heavy.

$$\texttt{HeavyOracle}(e,\alpha,\varepsilon) = \begin{cases} 1 & \text{if edge } e \text{ is a } \dfrac{h\alpha}{\varepsilon}\text{-heavy edge, i.e., } \tau = \dfrac{h\alpha}{\varepsilon} \\ 0 & \text{if edge } e \text{ is a } \dfrac{l\alpha}{\varepsilon}\text{-light edge, i.e., } \tau = \dfrac{l\alpha}{\varepsilon} \end{cases}$$

Here h and l ($h > l$) are constants to be determined later. We will discuss an efficient implementation of this oracle later.

Let us first consider the case where we are given an oracle that given an edge e, returns the exact value of a weight function, $w(e)$. Given we can sample edges uniformly at random through the `RandomEdge` query, we can compute $\mathbb{E}_{e\sim \mathrm{Unif}(E)}[w(e)] = T_L^{\frac{l\alpha}{\varepsilon}}$ using this oracle on the sampled edges. However, no such oracle exists in our model. Hence, we try to simulate one through an empirical estimate of the weight function, $\widehat{w}(e)$. Recall the fact that there are many valid weight functions, each being characterized by every light triangle being charged to one of its constituent light edges. Our algorithm will work if the empirical weight function $\widehat{w}$ estimates any one of these weight functions. We achieve this by ensuring that a triangle is not sampled through more than one edge (see Line 17 of Algorithm 1). Thus, the assignments that we consider can be extended to a valid weight function w, and our algorithm can be thought of as estimating this weight function through $\widehat{w}$. We develop our initial algorithm (Algorithm 1) assuming $\widetilde{T}$, an estimate of T satisfying $\widetilde{T} \le 2T$.

In Algorithm 1, we assume that we know m exactly. In fact, our algorithm and analysis work even if we have a constant factor approximation of m. This can be achieved by using $\widetilde{O}_{\varepsilon,\delta}(1)$ queries [7].

We now state and analyse the algorithm.

▶ **Lemma 14** (Expectation of Algorithm 1)**.** *Algorithm 1 ensures that $\mathbb{E}[\widehat{w}(e_i)] = T_L^{\frac{l\alpha}{\varepsilon}}/m$, and $\mathbb{E}\left[\widehat{T}\right] = T_L^{\frac{l\alpha}{\varepsilon}}$.*

Proof. Let $\mathcal{E}_i$ be the event that the edge e is chosen as the i-th edge sample. We proceed with the proof by considering two different cases: $\deg_e < \alpha$ and $\deg_e \ge \alpha$.

Algorithm 1 Triangle Counting Algorithm - with access to `Heavy`, and knowledge of $\widetilde{T}$.

Require: `Degree`, `Neighbour`, `Pair`, and `RandomEdge` query access to a graph G. Parameters $\widetilde{T}, \alpha, \varepsilon, m$ and oracle access to `HeavyOracle` with threshold constants l, h

$s \leftarrow c(1+h)\varepsilon^{-3}(m\alpha/\widetilde{T})\log n$
$S \leftarrow \emptyset$ ▷ S is the set of random edges sampled. $|S|$ grows up to s
$S_\Delta \leftarrow \emptyset$ ▷ S_Δ is the set of light triangles sampled through the edges in S
for $i \in [s]$ **do**
  $e_i \leftarrow$ `RandomEdge`
  $S \leftarrow S \cup e_i$
  Let $e_i = (v_i, x)$ where $\deg_{v_i} < \deg_x$ ▷ Requires two `Degree` queries
  if (`HeavyOracle`$(e_i, \alpha, \varepsilon) = 0$) **then** ▷ e_i is a $\frac{l\alpha}{\varepsilon}$-light edge
    $r_{e_i} \leftarrow 0$ ▷ r_e denotes the number of queries for each edge e
    if $\deg_{v_i} \leq \alpha$ **then**
      set $r_{e_i} \leftarrow 1$ with probability $\frac{\deg_{v_i}}{\alpha}$
    else
      set $r_{e_i} \leftarrow \left\lceil \frac{\deg_{v_i}}{\alpha} \right\rceil$
    for $j \in [r_{e_i}]$ **do**
      Choose $k \leftarrow \text{Unif}\left(\{1, 2, ..., \deg_{v_i}\}\right)$
      $u \leftarrow$ `Neighbour`(v_i, k)
      if (`Pair`$(u, x) = 1$ and $\mathtt{t} = (\mathtt{u}, \mathtt{e_i})$ is not in S_Δ through another edge e') **then**
        $S_\Delta \leftarrow S_\Delta \cup \mathtt{t}$
  ▷ A triangle $\mathtt{t}$ may occur in S_Δ multiple times, but each time it will be through the same edge e
  else
    $r_{e_i} \leftarrow 0$
for $i \in [s]$ **do**
  if $r_{e_i} > 0$ **then**
    $\widehat{w}(e_i) = \frac{1}{r_{e_i}} \sum_{(\mathtt{t}, \mathtt{e_i}) \in \mathtt{S}_\Delta} \max\left(\alpha, \deg_{e_i}\right)$
  else
    $\widehat{w}\left(e_i\right) = 0$
return $\widehat{T} = \frac{m}{s} \sum_{i \in s} \widehat{w}(e_i)$

Case I: ($\deg_e < \alpha$). When $\deg_e < \alpha$, r_e is set to 0 with probability $1 - \deg_e/\alpha$ and 1 with probability $\deg_e/\alpha$, and $\widehat{w}(e)$ is evaluated through a single query. Thus,

$$\mathbb{E}[\widehat{w}(e_i)|\mathcal{E}_i] = \frac{\deg_e}{\alpha} \sum_{k \in [w(e)]} \frac{1}{\deg_e}\alpha + \left(1 - \frac{\deg_e}{\alpha}\right) \cdot 0 = w(e) \tag{1}$$

Case II: ($\deg_e \geq \alpha$). On the other hand, when $\deg_e \geq \alpha$, let $Z_j, j \in [r_e]$ be the contribution of each r_e queries made for the edge e to the weight function estimate. Then,

$$\mathbb{E}[Z_j|\mathcal{E}_i] = \sum_{k \in [w(e)]} \frac{1}{\deg_e} \deg_e = w(e) \tag{2}$$

Correspondingly, we have by linearity of expectation and Equation (2):

$$\mathbb{E}\left[\widehat{w}(e_i)|\mathcal{E}_i\right] = \frac{1}{r_e}\sum_{j\in[r_e]}\mathbb{E}\left[Z_j|\mathcal{E}_i\right] = \mathbb{E}\left[Z_j|\mathcal{E}_i\right] = w(e) \tag{3}$$

Given that we draw each edge $e \in S$ uniformly at random, we now have by Equations (1), (3), and Lemma 13:

$$\mathbb{E}\left[\widehat{w}(e_i)\right] = \mathop{\mathbb{E}}_{e\sim\mathrm{Unif}(E)} w\left(e\right) = T_L^{\frac{l\alpha}{\varepsilon}}/m$$

By linearity of expectations, we obtain

$$\mathbb{E}\left[\widehat{T}\right] = \mathbb{E}\left[\frac{m}{s}\sum_{i\in s}\widehat{w}(e_i)\right] = T_L^{\frac{l\alpha}{\varepsilon}}$$ ◀

Note that Lemma 14 holds irrespective of whether the assumption $\widetilde{T} \leq 2T$ is satisfied or not. Next, we turn our attention to the variance of $\widehat{w}(e)$.

▶ **Lemma 15** (Variance of Algorithm 1). *Algorithm 1 ensures that* $\mathrm{Var}[\widehat{w}(e_i)] \leq \frac{(1+h)\alpha}{\varepsilon} \cdot \mathbb{E}_{e\sim\mathrm{Unif}(E)}[w(e)]$.

Proof. Again, we consider two different cases as in the proof of Lemma 14.

Case I: ($\deg_e < \alpha$). When $\deg_e < \alpha$, Algorithm 1 makes at most 1 query as $r \leq 1$. As $\widehat{w}(e) \leq \alpha$, we have

$$\mathrm{Var}[\widehat{w}(e_i)|\mathcal{E}_i] \leq \mathbb{E}[\widehat{w}(e_i)^2|\mathcal{E}_i] \leq \alpha\,\mathbb{E}\left[\widehat{w}\left(e_i\right)|\mathcal{E}_i\right] \tag{4}$$

Case II: ($\deg_e \geq \alpha$). Now we consider the case $\deg_e \geq \alpha$ which is more involved as $r \geq 1$. We first estimate the variance of each Z_j individually as defined in Lemma 14. We condition on the event that edge e is chosen as the i-th edge sample, denoted by $\mathcal{E}_i$. If $\texttt{HeavyOracle}(e,\alpha,\varepsilon) = 1$, then $Var[Z_j] = 0, \forall j \in [r]$. Hence, we condition on the event that $\texttt{HeavyOracle}(e,\alpha,\varepsilon) = 0$ from now on. As $Z_j \leq \deg_e$,

$$\mathrm{Var}[Z_j|\mathcal{E}_i] \leq \mathbb{E}[Z_j^2|\mathcal{E}_i] \leq \deg_{e_i}\mathbb{E}[Z_j|\mathcal{E}_i] \tag{5}$$

After having bounded the variance of the contribution of each query, we now obtain the variance of the weight estimate of the edge.

$$\begin{aligned}
\mathrm{Var}[\widehat{w}(e_i)|\mathcal{E}_i] &= \mathrm{Var}\left[\frac{1}{r_{e_i}}\sum_{j\in r_{e_i}} Z_j|\mathcal{E}_i\right] \\
&= \frac{1}{r_{e_i}^2}\sum_{j\in r_{e_i}}\mathrm{Var}\left[Z_j|\mathcal{E}_i\right] && (Z_j \text{ are i.i.d. given } \mathcal{E}_i) \\
&= \frac{\deg_{e_i}}{r_{e_i}}\sum_{j\in r_{e_i}}\frac{1}{r_{e_i}}\mathbb{E}[Z_j|\mathcal{E}_i] && (\text{by Equation (5) and linearity of expectation}) \\
&\leq \alpha\,\mathbb{E}[\widehat{w}(e_i)|\mathcal{E}_i] && \left(\text{as } r_{e_i} = \left\lceil\frac{\deg_e}{\alpha}\right\rceil\right)
\end{aligned} \tag{6}$$

Now, we remove the conditioning on $\mathcal{E}_i$ using the law of total variance:

$$\begin{aligned}
\mathrm{Var}[\widehat{w}(e_i)] &= \mathop{\mathbb{E}}_{e_i}[\mathrm{Var}[\widehat{w}(e_i)|\mathcal{E}_i]] + \mathop{\mathrm{Var}}_{e_i}[\mathbb{E}[\widehat{w}(e_i)|\mathcal{E}_i]] && \text{(by the law of total variance)}\\
&\leq \mathop{\mathbb{E}}_{e_i}[\alpha\,\mathbb{E}[\widehat{w}(e_i)|\mathcal{E}_i]] + \mathop{\mathbb{E}}_{e_i}[\mathbb{E}[\widehat{w}(e_i)|\mathcal{E}_i]^2] && \text{(by Equations (4) and (6))}\\
&\leq \alpha \mathop{\mathbb{E}}_{e_i}[\mathbb{E}[\widehat{w}(e_i)|\mathcal{E}_i]] + \frac{h\alpha}{\varepsilon}\cdot\mathop{\mathbb{E}}_{e_i}[\mathbb{E}[\widehat{w}(e_i)|\mathcal{E}_i]] && \text{(as } \texttt{HeavyOracle}(e)=0,\ \frac{h\alpha}{\varepsilon}\geq|T_e|)\\
&\leq \frac{(1+h)\alpha}{\varepsilon}\cdot\mathbb{E}[w(e)] && \blacktriangleleft
\end{aligned}$$

▶ **Theorem 16** (Performance guarantees of Algorithm 1). *Algorithm 1 makes* $c(1+h)\varepsilon^{-3}(m\alpha/\widetilde{T})\log n$ *queries, and* $c(1+h)\varepsilon^{-3}(m\alpha/\widetilde{T})\log n$ *calls to* `HeavyOracle`*, and given* $\widetilde{T}$ *satisfying the assumption that* $\widetilde{T}\leq 2T$*, returns* $\widehat{T}$ *such that* $\Pr(|\widehat{T}-T_L^{\frac{l\alpha}{\varepsilon}}|\geq\varepsilon T_L^{\frac{l\alpha}{\varepsilon}})\leq\frac{1}{c\log n}$.

Proof. The algorithm calls `HeavyOracle` for each of the s edges in S, resulting in a total of $c(1+h)\varepsilon^{-3}(\log n)(m\alpha/\widetilde{T})$ calls.

The algorithm makes s `RandomEdge` queries, $2s$ `Degree` queries, and $2r_e$ `Neighbour` query for each edge $e\in S$. All edges in S are sampled uniformly at random from E. Hence, the expected number of `Neighbour` queries made are:

$$\mathop{\mathbb{E}}_{e\sim\mathrm{Unif}(E)}\left[\left\lceil\frac{\deg_e}{\alpha}\right\rceil\right]=\sum_{e\in E}\frac{1}{m}\left\lceil\frac{\deg_e}{\alpha}\right\rceil\leq\frac{1}{m}\sum_{e\in E}1+\frac{\deg_e}{\alpha}=1+\frac{1}{m}\sum_{e\in E}\frac{\deg_e}{\alpha}\leq 1+\frac{2m\alpha}{m\alpha}=3$$

The last inequality in the above step follows from Lemma 5. Hence, the algorithm makes at most $9s=c(1+h)\varepsilon^{-3}(m\alpha/\widetilde{T})\log n$ queries in expectation. Here the inequality is due to the fact that for a heavy edge e in S, $r_e=0$. Furthermore, we bound the variance of the estimate $\widehat{T}$ as:

$$\mathrm{Var}[\widehat{T}]=\mathrm{Var}\left[\frac{m}{s}\sum_{i\in s}\widehat{w}(e_i)\right]\leq\frac{(1+h)\alpha m^2\,\mathbb{E}[w(e)]}{\varepsilon s}\leq\frac{(1+h)m\alpha\,\mathbb{E}[\widehat{T}]}{\varepsilon s}.$$

The last two steps follow from Lemma 15 and the fact that $\mathbb{E}\left[\widehat{T}\right]=m\,\mathbb{E}\left[w(e)\right]$. We now use Chebyshev's inequality on $\widehat{T}$:

$$\begin{aligned}
\Pr\left(|\widehat{T}-\mathbb{E}[\widehat{T}]|\leq\varepsilon\,\mathbb{E}[\widehat{T}]\right)&\leq\frac{\mathrm{Var}[\widehat{T}]}{\varepsilon^2\,\mathbb{E}[\widehat{T}]^2} && \text{(Chebyshev's inequality)}\\
&\leq\frac{(1+h)m\alpha}{\varepsilon^3 s\,\mathbb{E}[\widehat{T}]} && \text{(by Lemma 15)}\\
&\leq\frac{(1+h)m\alpha}{\varepsilon^3\cdot 4c\,(1+h)\,\varepsilon^{-3}(m\alpha/\widetilde{T})\log n\cdot T_L^{\frac{l\alpha}{\varepsilon}}} && \text{(by Lemma 14)}\\
&\leq\frac{1}{c\log n} && \text{(as } T_L^{\frac{l\alpha}{\varepsilon}}>T/2\geq\widetilde{T}/4)
\end{aligned}$$

where the last inequality uses the fact $T/2\geq\widetilde{T}/4$ due to the assumption that $\widetilde{T}\leq 2T$. ◀

4.3 Implementing the oracle

Rather than the exact oracle (`HeavyOracle`) that we assumed in Algorithm 1, we would design an oracle (called `Heavy`) that given arboricity α, and parameters ε and δ, accepts edges participating in at most $\frac{\alpha}{2\varepsilon}$ triangles as light, and rejects edges participating in at least

$\frac{2\alpha}{\varepsilon}$ triangles as heavy, with probability at least $1-\delta$. The algorithm works by estimating the number of triangles each edge participates in. However, in this case, each `Neighbour` and `Pair` queries generate i.i.d. random variables denoting the existence of triangle through that neighbouring vertex. Thus, our analysis uses the multiplicative Chernoff bound to obtain high probability guarantees for each individual edge.

Algorithm 2 `Heavy`$(e, \alpha, \varepsilon, \delta)$.

Require: `Degree`, `Neighbour`, `Pair`, and `RandomEdge` query access to a graph G

$r \leftarrow \left\lceil \frac{16\varepsilon \deg_e}{\alpha} \log\left(\frac{1}{\delta}\right) \right\rceil$ ▷ r denotes the number of queries for each edge e
Let $e = (u, x)$ where $\deg_u < \deg_x$ ▷ Requires 2 `Degree` queries
$Y \leftarrow 0$
for $i \in [r]$ **do**
  Choose $k \leftarrow \text{Unif}(\{1, 2, ..., \deg_u\})$
  $v_i \leftarrow$ `Neighbour`(u, k) ▷ Requires 1 `Neighbour` query
  if `Pair`$(v_i, x) = 1$ **then** ▷ Requires 1 `Pair` query
    $Y_i \leftarrow 1$ ▷ Found triangle (v_i, e)
  else
    $Y_i \leftarrow 0$
  $Y \leftarrow Y + Y_i$
$Y \leftarrow \frac{Y}{r}$
if $Y \geq \frac{\alpha}{\varepsilon \cdot \deg_e}$ **then**
  return 1
else
  return 0

▶ **Lemma 17** (Estimation guarantee of `Heavy`). *The algorithm* `Heavy`$(e, \alpha, \varepsilon, \delta)$ *satisfies the following properties with probability at least* $1-\delta$*: (i) returns* 1 *edge* e *if it is* $\frac{2\alpha}{\varepsilon}$*-heavy; (ii) returns* 0 *edge* e *if it is* $\frac{\alpha}{2\varepsilon}$*-light.*

Proof. For (i), we only consider edges that have at least $\frac{2\alpha}{\varepsilon}$ triangles. In this case, the random variables Y_i (Y_i as in Algorithm 2; $Y_i = 1$ if v_i and e form a triangle; 0, otherwise) are i.i.d. Bernoulli random variables taking value 1 with probability at least $\frac{2\alpha}{\varepsilon \deg_e}$. Hence, we have the following using linearity of expectation: $\mathbb{E}[Y] = \mathbb{E}\left[\frac{1}{r}\sum_{i\in[r]} Y_i\right] = \mathbb{E}[Y_i] > \frac{2\alpha}{\varepsilon \deg_e}$

As $Y = \frac{1}{r}\sum_i Y_i$, we can upper bound the probability of the algorithm returning 0 for the edge e, by a multiplicative Chernoff bound (Lemma 7) as follows:

$$\begin{aligned}
\Pr\left[Y \leq \frac{\alpha}{\varepsilon \deg_e}\right] &\leq \Pr\left[Y \leq \left(1-\frac{1}{2}\right)\mathbb{E}[Y]\right] && \left(\mathbb{E}[Y] > \frac{2\alpha}{\varepsilon \deg_e}\right) \\
&\leq \exp\left(-\frac{r\,\mathbb{E}[Y]}{12}\right) && \text{(by multiplicative Chernoff bound)} \\
&\leq \exp\left(-\frac{r\alpha}{6\deg_e \varepsilon}\right) && \left(\mathbb{E}[Y] > \frac{2\alpha}{\varepsilon \deg_e}\right) \\
&\leq \delta && \left(r = \frac{16\varepsilon \deg_e}{\alpha}\log\left(\frac{1}{\delta}\right)\right)
\end{aligned}$$

For (ii), we do not have a lower bound on the probability of success ($Y_i = 1$) in general and hence we cannot obtain a lower bound on $\mathbb{E}[Y]$. Now, observe that for the edges that participate in very few triangles, the probability of finding triangles (i.e., getting $Y_i = 1$) is

low. However, such light edges can still tolerate a high approximation error to be accepted (as a light edge). To account for the trade-off between the lower bound on the probability of $Y_i = 1$ and the upper bound on the approximation factor, we divide the edges participating in at most $\frac{\alpha}{2\varepsilon}$ triangles into $\left\lceil \log(\frac{\alpha}{\varepsilon}) \right\rceil$ buckets with each bucket being defined as the set of edges $\mathcal{B}_k = \left\{ e | \frac{\alpha}{2^{k+1}\varepsilon} \leq T_e < \frac{\alpha}{2^k \varepsilon} \right\}$. For each of these buckets, observe that when `Heavy` is called for an edge belonging to the bucket, we have $\Pr[Y_i = 1] = \frac{\mathsf{t_e}}{\deg_e} \geq \frac{\alpha}{2^{k+1}\varepsilon \deg_e}$, and hence $\mathbb{E}[Y] = \mathbb{E}[Y_i] \geq \frac{\alpha}{2^{k+1}\varepsilon \deg_e}$.

On the other hand, we have $\mathbb{E}[Y] < \frac{\alpha}{2^k \varepsilon \deg_e}$, and hence, $\left(1 + 2^{k-1}\right)\mathbb{E}[Y] < \frac{\alpha}{\varepsilon \deg_e}$. Using these observations, we complete the proof by considering any e in these buckets:

$$
\begin{aligned}
\Pr\left[Y \geq \frac{\alpha}{\varepsilon \deg_e} \right] &\leq \Pr\left[Y \geq \left(1 + 2^{k-1}\right)\mathbb{E}[Y] \right] && \left(\left(1 + 2^{k-1}\right)\mathbb{E}[Y] < \frac{\alpha}{\varepsilon \deg_e} \right) \\
&\leq \exp\left(- \frac{2^{2k-2} r \, \mathbb{E}[Y]}{2 + 2^{k-1}} \right) && \text{(by multiplicative Chernoff bound)} \\
&\leq \exp\left(- \frac{2^{2k} r \, \mathbb{E}[Y]}{2^{k+3}} \right) && (k \geq 1) \\
&\leq \exp\left(- \frac{r\alpha}{16\varepsilon \deg_e} \right) && \left(\mathbb{E}[Y] > \frac{\alpha}{2^{k+1}\varepsilon \deg_e} \right) \\
&\leq \delta && \left(r = \frac{16\varepsilon \deg_e}{\alpha} \log\left(\frac{1}{\delta}\right) \right) \quad \blacktriangleleft
\end{aligned}
$$

Lemma 17 shows that Algorithm 2 can decide whether an edge is heavy or not with probability at least $1 - \delta$ using $\frac{16\varepsilon}{\alpha} \deg_e \log\left(\frac{1}{\delta}\right)$ iterations; making 2 `Degree`, 1 `Neighbour` and 1 `Pair` queries in each iteration. For an individual edge, $\deg_e$ can be at most $n - 1$, and hence the subroutine might contribute to additional queries for each edge. However, as part of Algorithm 1, `Heavy` is called on a set of edges drawn uniformly at random. Hence, in the following lemma, we bound the expected number of queries made by `Heavy` when called on edges that were drawn uniformly at random.

▶ **Lemma 18** (Query complexity of `Heavy`). *`Heavy`$(e, \alpha, \varepsilon, \delta)$ makes $c\varepsilon \log\left(\frac{1}{\delta}\right)$ queries in expectation on an edge $e \sim \mathrm{Unif}(E)$.*

Proof. For edge e, `Heavy`$(e, \alpha, \varepsilon, \delta)$ makes $\left\lceil \frac{16\varepsilon \deg_e}{\alpha} \log\left(\frac{1}{\delta}\right) \right\rceil$ iterations with each iteration making 4 queries. Hence, the expected number of queries is:

$$
\begin{aligned}
\mathop{\mathbb{E}}_{e \sim \mathrm{Unif}(E)} \left[4 \left\lceil \frac{16\varepsilon \deg_e}{\alpha} \log\left(\frac{1}{\delta}\right) \right\rceil \right] &\leq \frac{64\varepsilon}{\alpha} \log\left(\frac{1}{\delta}\right) \mathop{\mathbb{E}}_{e \sim \mathrm{Unif}(E)} [\deg_e] + 4 \\
&= \frac{64\varepsilon}{\alpha} \log\left(\frac{1}{\delta}\right) \frac{1}{m} \sum_{e \in E} \deg_e + 4 \\
&\leq \frac{64\varepsilon}{\alpha} \log\left(\frac{1}{\delta}\right) \frac{1}{m} 2m\alpha + 4 && \text{(by Lemma 5)} \\
&\leq 132\varepsilon \log\left(\frac{1}{\delta}\right) && \left(\varepsilon \log\left(\frac{1}{\delta}\right) \geq 1 \right) \quad \blacktriangleleft
\end{aligned}
$$

4.4 Finalizing the algorithm

In this section, we combine the algorithms discussed above to obtain our final algorithm. Our goal is to call `Heavy` with appropriate parameters instead of `HeavyOracle` so that the approximation error due to the heavy triangles not being counted remains small. We also derive the query complexity of the algorithm including the queries made through `Heavy`.

4.4.1 Algorithm with an estimate $\widetilde{T}$

Algorithm 3 Triangle Counting Algorithm - with $\widetilde{T}$.

Require: `Degree`, `Neighbour`, `Pair`, and `RandomEdge` query access to a graph G. Parameters $\widetilde{T}, \alpha, \varepsilon, m$

1: Call Algorithm 1 with parameters $\widetilde{T}, \alpha, \varepsilon/2$, and thresholds $l = 6$, and $h = 24$. We also implement the oracle $\texttt{HeavyOracle}(e, \alpha, \varepsilon)$ through $\texttt{Heavy}\left(e, \alpha, \varepsilon/6, \frac{1}{mn}\right)$. ▷
Due to Algorithm 1 being called with parameter $\varepsilon/2$ and the implementation of `Heavy` as stated, the threshold parameters are evaluated w.r.t. call to `Heavy` with parameter $l = 6$ and $h = 24$

▶ **Theorem 19** (Performance guarantees of Algorithm 3)**.** *Algorithm 3 makes at most* $c\varepsilon^{-3}(m\alpha/\widetilde{T})\log^2 n$ *queries in expectation and returns* $\widehat{T}$ *such that* $\Pr\left[\widehat{T} \in [(1-\varepsilon)T, (1+\varepsilon)T]\right] \geq 1 - \frac{1}{c\log n}$.

Proof. Algorithm 3 will make the same number of calls to `Heavy` as did Algorithm 1 to `HeavyOracle`. By Theorem 16, Algorithm 3 makes $c(1+h)\varepsilon^{-3}(m\alpha/\widetilde{T})\log n$ queries and $c(1+h)\varepsilon^{-3}\log(n)(m\alpha/\widetilde{T})$ calls to `Heavy`. By Lemma 18, each call to `Heavy` requires $c\varepsilon\log n$ queries in expectation. Combining with the fact that we have $h = 24$, the algorithm requires $(c + c\varepsilon\log n)\left(\varepsilon^{-3}(m\alpha/\widetilde{T})c\log n\right) \leq c\varepsilon^{-3}(m\alpha/\widetilde{T})\log^2 n$ queries in expectation.

By Lemma 17, we can use $\texttt{Heavy}\left(e, \alpha, \varepsilon/6, \frac{1}{mn}\right)$ as an implementation of the idealized $\texttt{HeavyOracle}(e, \alpha, \varepsilon)$. This ensures that a $\frac{6\alpha}{\varepsilon}$-light edge is accepted and a $\frac{24\alpha}{\varepsilon}$-heavy edge is rejected with probability at least $1 - \frac{1}{mn}$. By union bound, the algorithm accepts all $\frac{6\alpha}{\varepsilon}$-light edges and rejects all $\frac{24\alpha}{\varepsilon}$-heavy edges in S with probability at least $1 - \frac{1}{n}$. Hence, by Corollary 11 and Lemma 14, we have,

$$\mathbb{E}\left[\widehat{T}\right] = T_L^{\frac{6\alpha}{\varepsilon}} \in [(1-\varepsilon/2)T, T] \tag{7}$$

By Theorem 16, we have that:

$$\Pr\left[\widehat{T} \in \left[(1-\varepsilon/2)T_L^{\frac{6\alpha}{\varepsilon}}, (1+\varepsilon/2)T_L^{\frac{6\alpha}{\varepsilon}}\right]\right] \geq 1 - \frac{1}{c\log n} \tag{8}$$

Combining Equations (7) and (8) and using union bound on the event that all oracle calls were executed correctly, for large enough n:

$$\Pr\left[\widehat{T} \in [(1-\varepsilon)T, (1+\varepsilon)T]\right] \geq 1 - \frac{1}{c\log n}$$ ◀

4.4.2 Getting rid of the assumption on $\widetilde{T}$

In this section, we describe the steps to obtain Theorem 22 from Theorem 19. First, we remove the assumption on the knowledge of $\widetilde{T}$ for Algorithm 3. To achieve that, we search for an appropriate choice of $\widetilde{T}$ starting from $\bar{t} = n^3$ and halving it each time we fail to find a $\widetilde{T}$ satisfying the assumption that $\widetilde{T} \leq 2T$. We must also bound the probability of $\bar{t}$ deviating significantly below T. To ensure that, for each value of $\bar{t}$, we run Algorithm 3 using values of $\widetilde{T}$ as $n^3, n^3/2, ..., \bar{t}$.

First we bound the probability that the Algorithm 4 terminates at a choice of $\widetilde{T}$ that does not satisfy the assumption that $\widetilde{T} \leq 2T$.

Algorithm 4 Triangle Counting Algorithm.

Require: `Degree`, `Neighbour`, `Pair`, and `RandomEdge` query access to a graph G. Parameters α, ε, m

1: **for** $\bar{t}$ in $\left[n^3, n^3/2, ..., 1\right]$ **do**
2: **for** $\widetilde{T}$ in $\left[n^3, n^3/2, ..., \bar{t}\right]$ **do**
3: **for** $i \in [2\log(c\log(n))]$ **do**
4: $X_i \leftarrow$ Solution to Algorithm 3 with parameters $\widetilde{T}, \alpha, \varepsilon$
5: $X \leftarrow \min_i X_i$
6: **if** $X \geq \widetilde{T}$ **then**
7: **return** X

▶ **Lemma 20** (Termination bound of Algorithm 4). *When $\widetilde{T} > 2T$, the algorithm terminates with probability at most $\frac{1}{c\log^2 n}$.*

Proof. By Lemma 14, we have from Markov's inequality:

$$\Pr\left[\widehat{T} > 2T\right] \leq \frac{\mathbb{E}\left[\widehat{T}\right]}{2T} \leq \frac{1}{2}$$

Now, we have $\Pr\left[X \geq \widetilde{T}\right] \leq \Pr\left[\cap_i X_i \geq \widetilde{T}\right] \leq \frac{1}{c\log^2 n}$ ◀

▶ **Lemma 21** (Lower bound of $\bar{t}$ for Algorithm 4). *The algorithm reaches a value of $\bar{t} \leq T/2^k$ $(k \geq 1)$ with probability at most $\frac{1}{(c\log n)^k}$.*

Proof. For every such value of $\bar{t}$, we have a $\widetilde{T}$ in Line 2 of Algorithm 4 such that $\widetilde{T} \leq T/2$. By Theorem 19, the algorithm returns an estimate $X_i \geq T/2 \geq \widetilde{T}$ with probability at least $1 - \frac{1}{c\log n}$. Taking a union bound over possible values of $i \in [2\log(c\log n)]$, we have Algorithm 4 returning an X such that $X \geq T/2 \geq \widetilde{T}$ with probability at least $1 - \frac{1}{c\log n}$, and the algorithm terminates.

Hence, to get to a $\bar{t} \leq \frac{T}{2^k}$, Algorithm 4 must have failed to terminate for all previous values of $\bar{t}$, which happens with probability at most $\frac{1}{(c\log n)^k}$. ◀

Finally we combine Lemmas 20 and 21 to obtain the following theorem quantifying the estimation guarantees and query complexity of Algorithm 4. The following theorem formalizes the Theorem 2 of Section 1.1.

▶ **Theorem 22** (Upper bound - Formal). *There exists an algorithm that makes $\widetilde{O}\left(\frac{m\alpha\log\frac{1}{\delta}}{\varepsilon^3 T}\right)$ queries in expectation, and returns $X \in [(1-\varepsilon)T, (1+\varepsilon)T]$ with probability at least $1-\delta$.*

Proof. The algorithm runs through at most $\log^2 n$ values of $\widetilde{T}$ such that $\widetilde{T} > 2T$. By Lemma 20, the algorithm terminates in each such case with probability at most $1/c\log^2 n$. For each value of $\bar{t} \leq 2T$, by Theorem 19, the algorithm returns a wrong value with probability at most $1/c\log n$. There are at most $3\log n$ such values of $\bar{t}$. Taking an union bound and fixing the constant c appropriately, the algorithm returns a correct output with probability at least $5/6$.

For each $\bar{t}$ in Line 1 of Algorithm 4, by Theorem 19, the algorithm makes $\widetilde{O}\left((m\alpha/\varepsilon^3\bar{t})\right)$ queries. Hence, till $\bar{t} \leq T/2$, the algorithm makes $\widetilde{O}\left((m\alpha/\varepsilon^3 T)\right)$ queries. Additionally, by Lemma 21, the queries beyond $\bar{t} \leq T$ can be bounded as:

$$\sum_{i\in\log(n)} \frac{1}{(c\log n)^k} \cdot 2^k \cdot \widetilde{O}\left(\frac{m\alpha}{\varepsilon^3 T}\right) = \widetilde{O}\left(\frac{m\alpha}{\varepsilon^3 T}\right)$$

By the well-known median trick [19], the median of $O\left(\log\frac{1}{\delta}\right)$ independent runs of Algorithm 4 establishes the result. ◀

5 Lower bound

In this section, we prove a lower bound that accounts for ε, δ, and the arboricity α of the graph by extending the ideas proposed in [8]. Our lower bound almost matches our proposed upper bound stated in Theorem 22. We first state the problem Popcount Thresholding Problem (referred to as PTP) in the query framework:

▶ **Definition 23** ($\texttt{PTP}_{M,k,\gamma}$). *Given a string $x \in \{0,1\}^M$, the problem $\textit{PTP}_{M,k,\gamma}$ is to distinguish whether x is generated by i.i.d. samples from $\mathcal{D}_0$ or $\mathcal{D}_1$ defined as follows:*

- $\boldsymbol{\mathcal{D}_0}$ *: For all $i \in [M]$, x_i is set to 1 with probability $(1-2\gamma)\frac{k}{M}$, and set to 0, otherwise.*
- $\boldsymbol{\mathcal{D}_1}$ *: For all $i \in [M]$, x_i is set to 1 with probability $(1+2\gamma)\frac{k}{M}$, and set to 0, otherwise.*

The problem is to decide if x is generated from $\mathcal{D}_0$ or $\mathcal{D}_1$ by querying x at any of its M bits.

We state the following lemma [8] quantifying the query complexity of $\texttt{PTP}_{M,k,\gamma}$.

▶ **Lemma 24** (PTP Query Lower Bound [8]). *For any $\gamma \in (0,1/4)$, $\delta \in (0,1/100)$, and integers $M \geq 1$, $\log(1/\delta)\cdot 12/\gamma^2 \leq k \leq M/6$, $R_\delta(\textit{PTP}_{M,k,\gamma}) \geq \frac{M\log(1/4\delta)}{24\gamma^2 k}$ where $R_\delta(\textit{PTP}_{M,k,\gamma})$ is the randomized query complexity to decide $\textit{PTP}_{M,k,\gamma}$ problem with probability at least $1-\delta$.*

We also state the following lemma, establishing bounds on the ℓ_1-norm of x, which was partially proved in [8].

▶ **Lemma 25.** *In $\textit{PTP}_{M,k,\gamma}$, for any $\gamma \in (0,1/4)$, $\delta \in (0,1/100)$, and integers $M \geq 1$, $\log(1/\delta)\cdot 12/\gamma^2 \leq k \leq M/6$, let $\|x\|_1$ denote the number of 1's in the string x, then:*

(i) $\Pr[\|x\|_1 > (1-\gamma)\cdot k \,|\, x \sim \mathcal{D}_0] \leq \delta$;

(ii) $\Pr[\|x\|_1 < (1+\gamma)\cdot k \,|\, x \sim \mathcal{D}_1] \leq \delta$;

(iii) $\Pr[\|x\|_1 < (1-4\gamma)\cdot k] \leq \delta$;

In [8], (i) and (ii) have been shown using Chernoff bound. (iii) can also be shown by using Chernoff bound. We combine two results to obtain the result. First we state the lemma due to [8]:

▶ **Lemma 26** ([8]). *In $\textit{PTP}_{M,k,\gamma}$, for any $\gamma \in (0,1/4)$, $\delta \in (0,1/100)$, and integers $M \geq 1$, $\log(1/\delta)\cdot 12/\gamma^2 \leq k \leq M/6$,*

(i) $\Pr[\|x\|_1 > (1-\gamma)\cdot k \mid x \sim \mathcal{D}_0] \leq \delta$;

(ii) $\Pr[\|x\|_1 < (1+\gamma)\cdot k \mid x \sim \mathcal{D}_1] \leq \delta$

where $\|x\|_1$ denotes the number of 1's in the string x.

We introduce another lemma to lower bound the ℓ_1 norm of x for $\mathcal{D}_0$.

▶ **Lemma 27.** *In $\textit{PTP}_{M,k,\gamma}$, for any $\gamma \in (0,1/4)$, $\delta \in (0,1/100)$, and integers $M \geq 1$, $\log(1/\delta)\cdot 12/\gamma^2 \leq k \leq M/6$, $\Pr[\|x\|_1 < (1-4\gamma)\cdot k \mid x \sim \mathcal{D}_0] \leq \delta$*

Proof. By definition of $\mathcal{D}_0$ and linearity of expectations, we have:

$$\mathbb{E}[\|x\|_1] = \sum_{i\in[M]} \mathbb{E}[x_i] = (1-2\gamma)\,k$$

Now, given the independence of each x_i, we can use Chernoff bound (Lemma 7) to obtain:

$$\begin{aligned}
&\Pr\left[\|x\|_1 < (1-4\gamma)\,k\right] \\
&= \Pr\left[\|x\|_1 - \mathbb{E}\left[\|x\|_1\right] > \frac{2\gamma}{1-2\gamma}\,\mathbb{E}\left[\|x\|_1\right]\right] \\
&\leq \exp\left(-\frac{4\gamma^2\,\mathbb{E}\left[\|x\|_1\right]}{3\,(1-2\gamma)^2}\right) && \text{(by Chernoff bound)} \\
&\leq \exp\left(-\frac{48\gamma^2\log\left(1/\delta\right)}{3\gamma^2\,(1-2\gamma)}\right) && \left(\mathbb{E}\left[\|x\|_1\right] = (1-2\gamma)\,k, k \geq \log(1/\delta)\cdot 12/\gamma^2\right) \\
&\leq \delta && (\gamma < 1/4)\ \blacktriangleleft
\end{aligned}$$

Combining Lemma 26 and 27, we obtain the desired result.

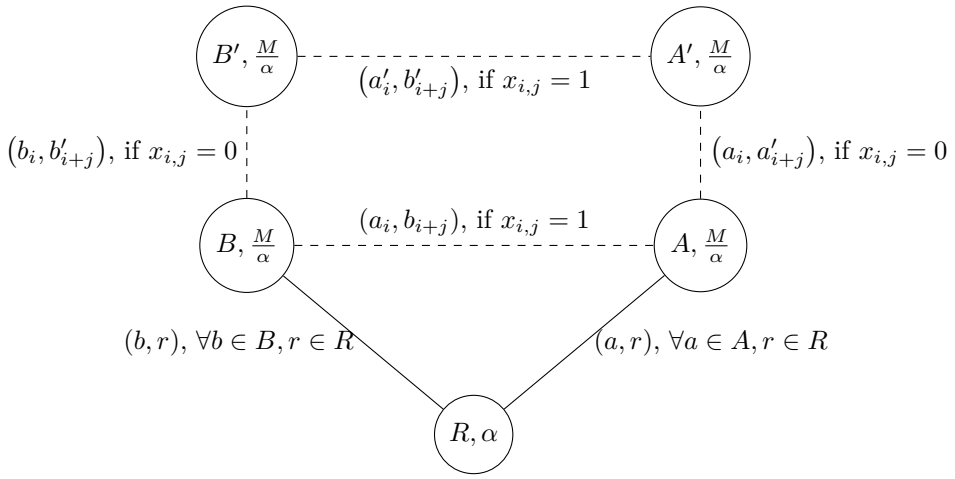


Figure 1 Lower Bound Graph Construction.

The following theorem formalizes the Theorem 3 of Section 1.1.

▶ **Theorem 28** (Lower bound - Formal). *Any algorithm that solves triangle estimation problem with $\varepsilon \leq \frac{1}{8}$ using* `Degree`, `Neighbour`, `Pair` *and* `RandomEdge` *for any graph with m edges, T triangles, and arboricity α such that $T = \Omega\left(\frac{\alpha\log(1/\delta)}{\varepsilon^2}\right)$ requires $\Omega\left(\frac{m\alpha\log(1/\delta)}{\varepsilon^2 T}\right)$ queries.*

Proof. We want to show that given a graph G, it takes $\Omega\left(\frac{m\alpha\log(1/\delta)}{\varepsilon^2 T}\right)$ queries to obtain an estimate $\widetilde{T}$ such that $\widetilde{T} \in (1\pm\varepsilon)T$ with probability at least $1-\delta$. For a contradiction, let us assume that there exists an algorithm $\mathcal{A}_T$ that, for some arboricity α, computes an estimate $\widetilde{T}$ such that $\widetilde{T} \in (1\pm\varepsilon)T$ with probability at least $1-\delta$ using $o\left(\frac{m\alpha\log(1/\delta)}{\varepsilon^2 T}\right)$ queries.

To show the contradiction, we design an algorithm that solves $\mathtt{PTP}_{M,k,\gamma}$ in less than $\frac{M\log(1/4\delta)}{24\gamma^2 k}$ queries using $\mathcal{A}_T$. Given an instance of x generated from $\mathtt{PTP}_{M,k,\gamma}$, we consider a graph $G_x = (V_x, E_x)$ defined as:

The vertex set V_x. The vertex set V_x consists of 5 sets of disjoint and independent set of vertices A, A', B, B', and R, with $|A| = |A'| = |B| = |B'| = M/\alpha$, and $|R| = \alpha$. Vertices of A, A', B and B' will be named as a, a', b and b', respectively.

The edge set E_x. There exists an edge between every vertex in $A \cup B$ to every vertex in R. Observe that this makes sure that the arboricity of G_x is at least the average degree of the subgraph induced by $A \cup B \cup R$, i.e., $\Omega(\alpha)$. If a bit of x is 1 (resp. 0), there will be an edge from a vertex in A (resp. A) to a vertex in B (resp. A'), and an edge from a vertex in A' (resp. B) to a vertex in B' (resp. B').

We index the problem instance x of length M as $\left[\frac{M}{\alpha}\right] \times [\alpha]$, denoting $x_{(i-1)\times\alpha+j}$ as $x_{i,j}$, where $i \in \left[\frac{M}{\alpha}\right]$ and $j \in [\alpha]$. We add an edge $\left(a_i, a'_{i+j}\right)$, and an edge $\left(b_i, b'_{i+j}\right)$ if $x_{i,j} = 0$. We add an edge (a_i, b_{i+j}), and an edge $\left(a'_i, b'_{i+j}\right)$ if $x_{i,j} = 1$. All $(i+j)$ additions here are modulo $\left[\frac{M}{\alpha}\right]$.

Consider the orientation of edges such that all edges from R is oriented outwards, and other edges are oriented arbitrarily. This gives a DAG with the out-degree of each edge being bounded by 2α. As a DAG with out-degree at most α ensures that the arboricity of the graph G_x is at most α[28, 47, 21], the graph has arboricity at most 2α. Hence, the arboricity of G_x is $\Theta(\alpha)$, as we have already shown it to be $\Omega(\alpha)$.

From the construction, observe that the number of edges between A and B is $\|x\|_1$. Moreover, each edge between A and B adds α triangles. Hence, the number of triangles in G_x is exactly $\|x\|_1 \alpha$. Also note that we have added $2M$ edges between $A \cup B$ and R, and further $2M$ edges (in the subgraph induced by $V_x \setminus S$) according to the entries of x, 2 for each bit x_i. Hence, we have $m = 4M$, and we fix $\varepsilon = \gamma$.

We now describe $\mathcal{A}_T^{\mathtt{PTP}}$ an algorithm that uses $\mathcal{A}_T$ to solve $\mathtt{PTP}_{M,k,\gamma}$ using less than $\frac{M \log(1/4\delta)}{24\gamma^2 k}$ queries. Given a string x, we generate G_x and estimate the number of triangles $\widehat{T}$ by calling $\mathcal{A}_T$ using $\frac{m \log(1/4\delta)}{200\varepsilon^2 k} = \frac{M \log(1/4\delta)}{50\gamma^2 k}$ queries. If $\widehat{T} < \left(1-\gamma^2\right) k\alpha$, it outputs $x \sim \mathcal{D}_0$; otherwise, it outputs $x \sim \mathcal{D}_1$. Also, observe that by Lemma 25, we have $T = \alpha \|x\|_1 \geq \alpha(1-4\gamma)k \geq \frac{\alpha k}{2}$ with probability at least $1-\delta$, where the last inequality follows from the fact that $\gamma = \varepsilon \leq \frac{1}{4}$. Lemma 25 is applicable due to the fact that $T = \Omega\left(\frac{\alpha \log(1/\delta)}{\varepsilon^2}\right)$. Hence, we have $\frac{m \log(1/4\delta)}{200\varepsilon^2 k} \geq \frac{m\alpha \log(1/4\delta)}{200\varepsilon^2 T}$. Thus, $\mathcal{A}_T$ is allowed to make $o\left(\frac{m\alpha \log(1/\delta)}{\varepsilon^2 T}\right)$ queries under the given query bound.

By Lemma 25, to distinguish between whether x is generated from $\mathcal{D}_0$ or $\mathcal{D}_1$ with probability at least $1-\delta$, it suffices to show that we can use $\mathcal{A}_T$ to distinguish between the instance where $\|x\|_1 > (1+\gamma)\,k$ and the instance where $\|x\|_1 < (1-\gamma)\,k$ instance with probability at least $1-\frac{\delta}{2}$. We now show that $\mathcal{A}_T^{\mathtt{PTP}}$ outputs $\mathcal{D}_1$ with probability at least $1-\frac{\delta}{2}$ given $\|x\|_1 > (1+\gamma)\,k$, and outputs $\mathcal{D}_0$ with probability at least $1-\frac{\delta}{2}$ given $\|x\|_1 < (1-\gamma)\,k$. We consider the two cases separately:

Case I: $(\|x\|_1 > (1+\gamma)\,k)$. Our construction ensures that the number of triangles in G_x is $T = \|x\|_1 \alpha > (1+\gamma)\,k\alpha = (1+\varepsilon)\,k\alpha$. Additionally, by our assumption on $\mathcal{A}_T$, it outputs an estimate $\widehat{T} \geq (1-\varepsilon)T$ with probability $1-\frac{\delta}{2}$. Thus, we have $\widetilde{T} \geq \left(1-\varepsilon^2\right) k\alpha$ with probability at least $1-\frac{\delta}{2}$.

Case II: $(\|x\|_1 < (1-\gamma)\,k)$. Our construction ensures that the number of triangles in G_x is $T = \|x\|_1 \alpha < (1-\gamma)\,k\alpha = (1-\varepsilon)\,k\alpha$. Additionally, by our assumption on $\mathcal{A}_T$, it outputs an estimate $\widehat{T} \leq (1+\varepsilon)T$ with probability $1-\frac{\delta}{2}$. Thus, we have $\widetilde{T} \leq \left(1-\varepsilon^2\right) k\alpha$ with probability at least $1-\frac{\delta}{2}$.

Now we state how to simulate the required queries used by the algorithm:

- **`Degree`(v):** For a vertex $v \in A \cup B \cup A' \cup B'$, return α, and for a vertex $v \in R$, return $\frac{2M}{\alpha}$. Hence, for `Degree` queries, no queries are made to the string.
- **`Neighbour`(v, i):** For a vertex $v \in A \cup B \cup A' \cup B'$, w.l.o.g assume v to be $a_j \in A$. Return a'_{i+j} if $x_{i,j} = 0$, and b_{i+j} otherwise. For a vertex $v \in R$, if $i \leq \frac{m}{\alpha}$, return a_i, else return $b_{i-\frac{m}{\alpha}}$. Hence, for `Neighbour` queries, a single query is made to the string.

- **`Pair`(u, v):** Given a vertex pair (v, u), there can be 3 cases: (a) If the vertex $v \in A \cup B$, w.l.o.g assume v to be $a_i \in A$, if $u \in R$, return 1, else if $u = b_j \in B$ and $x_{i,j-i} = 1$, return 1, else if $u = a'_j \in A'$ and $x_{i,j-i} = 0$, return 1, else return 0. (b) If a vertex $v \in A' \cup B'$, w.l.o.g assume the v to be $a'_i \in A'$, if $u = a_j \in A$, return 1 if $x_{j,i-j} = 0$, else if $u = b'_j \in B'$, return 1 if $x_{j,i-j} = 1$, else return 0. If $v \in R$, return 1 if $u \in A \cup B$, else return 0. (c) If a vertex $v \in R$, return 1 if $u \in A \cup B$, return 0 otherwise. Hence, for `Pair` queries, a single query is made to the string.
- **`RandomEdge`:** Sample a vertex v with probability $\frac{\deg_v}{4M}$. Choose $i \in \deg_{(v)}$ uniformly at random, query $u =$ `Neighbour`(v, i) and return (v, u). Hence, for `RandomEdge` queries, a single query is made to the string for the `Neighbour` query. ◀

The following corollary is a direct implication from Theorem 28 due to the fact that we consider a weaker model (without access to `RandomEdge` queries) compared to Theorem 28.

▶ **Corollary 29** (Lower Bound for Local Queries). *Any algorithm that solves triangle estimation problem with $\varepsilon \leq \frac{1}{4}$ using `Degree`, `Neighbour`, and `Pair` requires $\Omega\left(\frac{m\alpha \log(1/\delta)}{\varepsilon^2 T}\right)$ queries.*

6 Conclusion

In this paper, we present an almost optimal algorithm for triangle counting in graphs of bounded arboricity, assuming access to both local queries and `RandomEdge` queries. In particular, the query complexity of our algorithm is $O(m\alpha/T)$.[3] Extending our result to clique counting remains an interesting open question. Eden et al. [28] showed that the number of k-cliques can be estimated using $O\left(n/T_k^{1/k} + m^{k/2}/T_k\right)$ local queries, where T_k denotes the number of k-cliques in the graph. When `RandomEdge` queries are also available, Assadi et al. [7] established a bound of $O\left(m^{k/2}/T_k\right)$. For graphs with arboricity bounded by α, Eden et al. [28] showed that $O\left(n/T_k^{1/k} + m\alpha^{k-2}/T_k\right)$ local queries suffice. We believe that, when `RandomEdge` queries are available in addition to local queries, the bound can be improved to $O\left(m\alpha^{k-2}/T_k\right)$. However, establishing the complexity of clique counting (when `RandomEdge` is available along with local queries) remains an open question. A similar open question was posed in the streaming model by Bera and Seshadri [13].

References

1 Charu Aggarwal and Karthik Subbian. Evolutionary network analysis: A survey. *ACM Comput. Surv.*, 47(1), May 2014. doi:10.1145/2601412.

2 Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Graph sketches: sparsification, spanners, and subgraphs. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS '12, pages 5–14, New York, NY, USA, 2012. Association for Computing Machinery. doi:10.1145/2213556.2213560.

3 Mohammad Al Hasan and Vachik S. Dave. Triangle counting in large networks: a review. *WIREs Data Mining and Knowledge Discovery*, 8(2):e1226, 2018. doi:10.1002/widm.1226.

4 N. Alon. Testing Subgraphs in Large Graphs. In *IEEE 54th Annual Symposium on Foundations of Computer Science*, page 434, Los Alamitos, CA, USA, October 2013. IEEE Computer Society. doi:10.1109/SFCS.2001.959918.

5 Noga Alon, Tali Kaufman, Michael Krivelevich, and Dana Ron. Testing triangle-freeness in general graphs. *SIAM J. Discret. Math.*, 22(2):786–819, 2008. doi:10.1137/07067917X.

[3] In this section, we suppress the dependence on polylogarithmic factors, for simplicity of presentation.

6 Noga Alon, Raphael Yuster, and Uri Zwick. Finding and counting given length cycles. *Algorithmica*, 17(3):209–223, 1997. doi:10.1007/BF02523189.

7 Sepehr Assadi, Michael Kapralov, and Sanjeev Khanna. A Simple Sublinear-Time Algorithm for Counting Arbitrary Subgraphs via Edge Sampling. In Avrim Blum, editor, *10th Innovations in Theoretical Computer Science Conference (ITCS 2019)*, volume 124 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 6:1–6:20, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2019.6.

8 Sepehr Assadi and Hoai-An Nguyen. Asymptotically optimal bounds for estimating h-index in sublinear time with applications to subgraph counting. In Amit Chakrabarti and Chaitanya Swamy, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2022, September 19-21, 2022, University of Illinois, Urbana-Champaign, USA (Virtual Conference)*, volume 245 of *LIPIcs*, pages 48:1–48:20, Illinois, Urbana-Champaign, USA (Virtual Conference), 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX/RANDOM.2022.48.

9 Albert Atserias, Martin Grohe, and Dániel Marx. Size bounds and query plans for relational joins. *SIAM Journal on Computing*, 42(4):1737–1767, 2013. doi:10.1137/110859440.

10 Ziv Bar-Yossef, Ravi Kumar, and D. Sivakumar. Reductions in streaming algorithms, with an application to counting triangles in graphs. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '02, pages 623–632, USA, 2002. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=545381.545464.

11 Suman K. Bera and Amit Chakrabarti. Towards tighter space bounds for counting triangles and other substructures in graph streams. In Heribert Vollmer and Brigitte Vallée, editors, *34th Symposium on Theoretical Aspects of Computer Science, STACS 2017, March 8-11, 2017, Hannover, Germany*, volume 66 of *LIPIcs*, pages 11:1–11:14, Hannover, Germany, 2017. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.STACS.2017.11.

12 Suman K. Bera, Amit Chakrabarti, and Prantar Ghosh. Graph Coloring via Degeneracy in Streaming and Other Space-Conscious Models. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*, volume 168 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:21, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2020.11.

13 Suman K. Bera and C. Seshadhri. How the degeneracy helps for triangle counting in graph streams. In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS'20, pages 457–467, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3375395.3387665.

14 Lorenzo Beretta and Jakub Tětek. Better sum estimation via weighted sampling. *ACM Trans. Algorithms*, 20(3), June 2024. doi:10.1145/3650030.

15 Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. *computational complexity*, 21(2):311–358, May 2012. doi:10.1007/s00037-012-0040-x.

16 Vladimir Braverman, Rafail Ostrovsky, and Dan Vilenchik. How hard is counting triangles in the streaming model? In *40th International Colloquium on Automata, Languages, and Programming (ICALP 2013)*, ICALP'13, pages 244–254, Berlin, Heidelberg, 2013. Springer-Verlag. doi:10.1007/978-3-642-39206-1_21.

17 Laurent Bulteau, Vincent Froese, Konstantin Kutzkov, and Rasmus Pagh. Triangle counting in dynamic graph streams. *Algorithmica*, 76(1):259–278, September 2016. doi:10.1007/s00453-015-0036-4.

18 Luciana S. Buriol, Gereon Frahling, Stefano Leonardi, Alberto Marchetti-Spaccamela, and Christian Sohler. Counting triangles in data streams. In *Proceedings of the Twenty-Fifth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '06, pages 253–262, New York, NY, USA, 2006. Association for Computing Machinery. doi:10.1145/1142351.1142388.

19 Amit Chakrabarti. Data stream algorithms. *Computer Science*, 49:149, 2015.

20 Norishige Chiba and Takao Nishizeki. Arboricity and subgraph listing algorithms. *SIAM J. Comput.*, 14(1):210–223, 1985. doi:10.1137/0214017.

21 Marek Chrobak and David Eppstein. Planar orientations with low out-degree and compaction of adjacency matrices. *Theoretical Computer Science*, 86(2):243–266, 1991. doi:10.1016/0304-3975(91)90020-3.

22 Graham Cormode and Hossein Jowhari. A second look at counting triangles in graph streams (corrected). *Theoretical Computer Science*, 683:22–30, 2017. doi:10.1016/j.tcs.2016.06.020.

23 Maximilien Danisch, Oana Balalau, and Mauro Sozio. Listing k-cliques in sparse real-world graphs*. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, pages 589–598, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee. doi:10.1145/3178876.3186125.

24 Michal Dory, Mohsen Ghaffari, and Saeed Ilchi. Near-optimal distributed dominating set in bounded arboricity graphs. In *Proceedings of the 2022 ACM Symposium on Principles of Distributed Computing*, PODC'22, pages 292–300, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3519270.3538437.

25 Jean-Pierre Eckmann and Elisha Moses. Curvature of co-links uncovers hidden thematic layers in the world wide web. *Proceedings of the National Academy of Sciences of the United States of America*, 99:5825–5829, 2001. URL: https://api.semanticscholar.org/CorpusID:2618949.

26 Talya Eden, Amit Levi, Dana Ron, and C. Seshadhri. Approximately counting triangles in sublinear time. *SIAM Journal on Computing*, 46(5):1603–1646, 2017. doi:10.1137/15M1054389.

27 Talya Eden, Dana Ron, and C. Seshadhri. On approximating the number of k-cliques in sublinear time. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 722–734, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3188745.3188810.

28 Talya Eden, Dana Ron, and C. Seshadhri. Faster sublinear approximation of the number of k-cliques in low-arboricity graphs. In *Proceedings of the Thirty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '20, pages 1467–1478, USA, 2020. Society for Industrial and Applied Mathematics. doi:10.1137/1.9781611975994.89.

29 Talya Eden and Will Rosenbaum. Lower Bounds for Approximating Graph Parameters via Communication Complexity. In Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018)*, volume 116 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:18, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.11.

30 Irene Finocchi, Marco Finocchi, and Emanuele G. Fusco. Clique counting in mapreduce: Algorithms and experiments. *ACM J. Exp. Algorithmics*, 20, October 2015. doi:10.1145/2794080.

31 Gaurav Goel and Jens Gustedt. Bounded arboricity to determine the local structure of sparse graphs. In *Proceedings of the 32nd International Conference on Graph-Theoretic Concepts in Computer Science*, WG'06, pages 159–167, Berlin, Heidelberg, 2006. Springer-Verlag. doi:10.1007/11917496_15.

32 Oded Goldreich. *On the communication complexity methodology for proving lower bounds on the query complexity of property testing*, pages 87–118. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Springer Verlag, Germany, April 2020. doi:10.1007/978-3-030-43662-9_7.

33 Mira Gonen, Dana Ron, and Yuval Shavitt. Counting stars and other small subgraphs in sublinear-time. *SIAM Journal on Discrete Mathematics*, 25(3):1365–1411, 2011. doi:10.1137/100783066.

34 Shweta Jain and C. Seshadhri. A fast and provable method for estimating clique counts using turán's theorem. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 441–449, Republic and Canton of Geneva, CHE, 2017. International World Wide Web Conferences Steering Committee. doi:10.1145/3038912.3052636.

35 Rajesh Jayaram and John Kallaugher. An Optimal Algorithm for Triangle Counting in the Stream. In *APPROX/RANDOM 2021*, volume 207 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:11, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX/RANDOM.2021.11.

36 Madhav Jha, C. Seshadhri, and Ali Pinar. A space-efficient streaming algorithm for estimating transitivity and triangle counts using the birthday paradox. *ACM Trans. Knowl. Discov. Data*, 9(3), February 2015. doi:10.1145/2700395.

37 John Kallaugher, Michael Kapralov, and Eric Price. The sketching complexity of graph and hypergraph counting. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 556–567. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00059.

38 John Kallaugher, Andrew McGregor, Eric Price, and Sofya Vorotnikova. The complexity of counting cycles in the adjacency list streaming model. In Dan Suciu, Sebastian Skritek, and Christoph Koch, editors, *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, PODS 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, pages 119–133. ACM, 2019. doi:10.1145/3294052.3319706.

39 John Kallaugher and Eric Price. A hybrid sampling scheme for triangle counting. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, pages 1778–1797, USA, 2017. Society for Industrial and Applied Mathematics. doi:10.1137/1.9781611974782.116.

40 John Kallaugher and Eric Price. Separations and equivalences between turnstile streaming and linear sketching. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 1223–1236. ACM, 2020. doi:10.1145/3357713.3384278.

41 Daniel M. Kane, Kurt Mehlhorn, Thomas Sauerwald, and He Sun. Counting arbitrary subgraphs in data streams. In *Proceedings of the 39th International Colloquium Conference on Automata, Languages, and Programming - Volume Part II*, ICALP'12, pages 598–609, Berlin, Heidelberg, 2012. Springer-Verlag. doi:10.1007/978-3-642-31585-5_53.

42 Christian Konrad, Andrew McGregor, Rik Sengupta, and Cuong Than. Matchings in Low-Arboricity Graphs in the Dynamic Graph Stream Model. In Siddharth Barman and Sławomir Lasota, editors, *44th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2024)*, volume 323 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 29:1–29:15, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.FSTTCS.2024.29.

43 Tsvi Kopelowitz, Seth Pettie, and Ely Porat. Higher lower bounds from the 3sum conjecture. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, pages 1272–1287, USA, 2016. Society for Industrial and Applied Mathematics. doi:10.1137/1.9781611974331.CH89.

44 Jure Leskovec, Lars Backstrom, Ravi Kumar, and Andrew Tomkins. Microscopic evolution of social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 462–470, New York, NY, USA, 2008. Association for Computing Machinery. doi:10.1145/1401890.1401948.

45 Quanquan C. Liu and C. Seshadhri. Brief announcement: Improved massively parallel triangle counting in o(1) rounds. In *Proceedings of the 43rd ACM Symposium on Principles of Distributed Computing*, PODC '24, pages 519–522, New York, NY, USA, 2024. Association for Computing Machinery. doi:10.1145/3662158.3662819.

46 Duncan Luce and Albert D. Perry. A method of matrix analysis of group structure. *Psychometrika*, 14:95–116, 1949. URL: https://api.semanticscholar.org/CorpusID:16186758.

47 David W. Matula and Leland L. Beck. Smallest-last ordering and clustering and graph coloring algorithms. *J. ACM*, 30(3):417–427, July 1983. doi:10.1145/2402.322385.

48 Andrew McGregor, Sofya Vorotnikova, and Hoa T. Vu. Better algorithms for counting triangles in data streams. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS '16, pages 401–411, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2902251.2902283.

49 Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, Cambridge, 2005. doi:10.1017/CBO9780511813603.

50 Krzysztof Onak, Baruch Schieber, Shay Solomon, and Nicole Wein. Fully dynamic mis in uniformly sparse graphs. *ACM Trans. Algorithms*, 16(2), March 2020. doi:10.1145/3378025.

51 Rasmus Pagh and Charalampos E. Tsourakakis. Colorful triangle counting and a mapreduce implementation. *Inf. Process. Lett.*, 112(7):277–281, 2012. doi:10.1016/J.IPL.2011.12.007.

52 A. Pavan, Kanat Tangwongsan, Srikanta Tirthapura, and Kun-Lung Wu. Counting and sampling triangles from a graph stream. *Proc. VLDB Endow.*, 6(14):1870–1881, 2013. doi:10.14778/2556549.2556569.

53 Thomas Schank and Dorothea Wagner. Finding, counting and listing all triangles in large graphs, an experimental study. In Sotiris E. Nikoletseas, editor, *Experimental and Efficient Algorithms*, pages 606–609, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. doi:10.1007/11427186_54.

54 Kijung Shin, Tina Eliassi-Rad, and Christos Faloutsos. Patterns and anomalies in k-cores of real-world graphs with applications. *Knowl. Inf. Syst.*, 54(3):677–710, March 2018. doi:10.1007/s10115-017-1077-6.

55 Kijung Shin, Sejoon Oh, Jisu Kim, Bryan Hooi, and Christos Faloutsos. Fast, accurate and provable triangle counting in fully dynamic graph streams. *ACM Trans. Knowl. Discov. Data*, 14(2), February 2020. doi:10.1145/3375392.

56 Siddharth Suri and Sergei Vassilvitskii. Counting triangles and the curse of the last reducer. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 607–614, New York, NY, USA, 2011. Association for Computing Machinery. doi:10.1145/1963405.1963491.

57 Charalampos Tsourakakis, Mihail Kolountzakis, and Gary Miller. Triangle sparsifiers. *Journal of Graph Algorithms and Applications*, 15(6):703–726, October 2011. doi:10.7155/jgaa.00245.

58 Jakub Tětek. Approximate Triangle Counting via Sampling and Fast Matrix Multiplication. In Mikołaj Bojańczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, volume 229 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 107:1–107:20, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2022.107.

59 Virginia Vassilevska Williams and Yinzhan Xu. Monochromatic triangles, triangle listing and APSP. In Sandy Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 786–797, Durham,NC,USA, 2020. IEEE. doi:10.1109/FOCS46700.2020.00078.

60 Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998. doi:10.1038/30918.

An Improved Construction of Variety-Evasive Subspace Families

Robert Andrews
Cheriton School of Computer Science, University of Waterloo, Canada

Abhibhav Garg
Cheriton School of Computer Science, University of Waterloo, Canada

Abstract

We study the question of explicitly constructing variety-evasive subspace families, a pseudorandom primitive introduced by Guo (Computational Complexity 2024) that generalizes both hitting sets and lossless rank condensers. Roughly speaking, a variety-evasive subspace family $\mathcal{H}$ is a collection of subspaces such that for every algebraic variety V in a fixed family $\mathcal{F}$, there is some subspace $W \in \mathcal{H}$ that is in general position with respect to V.

We give an explicit construction of a subspace families that evade all degree-d varieties in an n-dimensional affine or projective space. Our construction improves on the size of the variety-evasive subspace families constructed by Guo and, for varieties of degree $n^{1+\Omega(1)}$, is polynomially close to Guo's lower bound on the size of any such variety-evasive subspace family. Our variety-evasive subspace families rely on an improved construction of hitting sets for Chow forms of algebraic varieties.

2012 ACM Subject Classification Theory of computation → Algebraic complexity theory; Theory of computation → Pseudorandomness and derandomization

Keywords and phrases algebraic complexity, pseudorandomness, varieties, variety-evasive subspaces, Chow forms

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.56

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2605.05621

1 Introduction

1.1 Background

The probabilistic method is a powerful tool with widespread use throughout mathematics. Many objects of interest throughout combinatorics and computer science, such as Ramsey graphs and randomness extractors, can be shown to exist via the probabilistic method, often with excellent parameters. While probabilistic arguments excel at demonstrating existence, they fall short of providing an efficient, deterministic means of constructing the object of interest. One of the main goals of pseudorandomness is to obtain explicit constructions of objects that exhibit properties similar to those enjoyed by a randomly-chosen object. Not only are explicit constructions interesting in their own right, but they are all the more important in algorithmic applications such as derandomization, where the aim is to reduce or eliminate the use of randomness as an algorithmic resource.

Over time, the theory of pseudorandomness has developed into a rich area of theoretical computer science that studies a variety of objects, including expander graphs, list-decodable codes, randomness extractors, pseudorandom generators, and the numerous connections and relationships between them [14]. More recently, a parallel theory of linear-algebraic pseudorandomness has also appeared. This theory studies collections of linear maps that


© Robert Andrews and Abhibhav Garg;
licensed under Creative Commons License CC-BY 4.0

satisfy various pseudorandom properties, including rank extractors and rank condensers, subspace designs, dimension expanders, and subspace-evasive sets, where the dimension of vector spaces plays a role analogous to that of min-entropy in classical pseudorandomness.

In this work, we study the construction of *variety-evasive subspace families*, a linear-algebraic pseudorandom object introduced by Guo [7]. Roughly speaking, a variety-evasive subspace family $\mathcal{H}$ is a collection of linear or affine spaces such that for any algebraic variety V of interest, there is some subspace $W \in \mathcal{H}$ that is in general position with respect to V. For example, if V is a curve in the plane and $\mathcal{H}$ is a collection of lines, then there should be some line $W \in \mathcal{H}$ such that the intersection $V \cap W$ is nonempty and zero-dimensional. More generally, when the variety V is fixed, most linear spaces (and in fact, most varieties) W will satisfy $\operatorname{codim}(V \cap W) = \operatorname{codim}(V) + \operatorname{codim}(W)$, and a variety-evasive subspace family must contain a subspace W that intersects V in a manner similar to a random subspace. This is the sort of behavior captured by the notion of one variety evading another, defined below.

▶ **Definition 1.1** (Evasiveness). *Let V and W be irreducible subvarieties of $\mathbb{P}^n$ or $\mathbb{A}^n$. We say that W* evades *V if*

$$\dim(V \cap W) \leq \dim(V) + \dim(W) - n.$$

If V and W are affine varieties, we say that W strongly evades *V if the above holds with equality whenever the right-hand side is nonnegative. If V is a reducible variety, we say that W* (strongly) evades *V if W (strongly) evades each irreducible component of V.*

When V and W are affine varieties, it may be the case that $V \cap W = \varnothing$, even though $\operatorname{codim}(V) + \operatorname{codim}(W)$ is not large enough to force this intersection to be empty in general. The notion of W strongly evading V adds the constraint that the intersection $V \cap W$ should have the expected codimension of $\operatorname{codim}(V) + \operatorname{codim}(W)$ whenever this is possible.

As a matter of notational convenience, we also define the notion of a k-subspace family.

▶ **Definition 1.2** (k-Subspace family). *For $0 \leq k \leq n$, a* k-subspace family *is a finite collection of k-dimensional subspaces of $\mathbb{P}^n$. Similarly, an* affine k-subspace family *is a finite collection of k-dimensional affine subspaces of $\mathbb{A}^n$.*

We can now define variety-evasive subspace families, the main object of study in this work.

▶ **Definition 1.3** (Variety-evasive subspace families [7]). *Let $\mathcal{F}$ be a family of subvarieties of $\mathbb{P}^n$ (or of $\mathbb{A}^n$) and let $\mathcal{H}$ be a k-subspace (or affine k-subspace) family.*

1. *We say that the family $\mathcal{H}$ is* (strongly) $\mathcal{F}$-evasive *if for every $V \in \mathcal{F}$, there is some $W \in \mathcal{H}$ that (strongly) evades V. In the case where $\mathcal{F}$ is the family of all degree-d subvarieties, we say that $\mathcal{H}$ is* (strongly) (n, d)-evasive.
2. *We say that $\mathcal{H}$ is* (strongly) $(\mathcal{F}, \varepsilon)$-evasive *if for every $V \in \mathcal{F}$, a randomly-chosen $W \in \mathcal{H}$ (strongly) evades V with probability at least $1 - \varepsilon$. When $\mathcal{F}$ is the family of all degree-d subvarieties, we say that $\mathcal{H}$ is* (strongly) (n, d, ε)-evasive.

As observed by Guo [7], variety-evasive subspace families are a common generalization of two natural problems in algebraic pseudorandomness. The first such problem is the construction of hitting sets for arithmetic circuits, a common approach to derandomizing algorithms for the polynomial identity testing problem. A hitting set $\mathcal{H} \subseteq \mathbb{F}^n$ is a collection of points such that for every nonzero polynomial $f \in \mathcal{C}$ in a circuit class $\mathcal{C}$ of interest, there is some point $\boldsymbol{\alpha} \in \mathcal{H}$ such that $f(\boldsymbol{\alpha}) \neq 0$. The point $\boldsymbol{\alpha}$ witnesses the fact that f is nonzero

as a polynomial. An efficient, deterministic construction of such a hitting set immediately implies a deterministic algorithm with similar complexity for testing polynomials $f \in \mathcal{C}$. Geometrically, one can view a hitting set as a set of points $\mathcal{H}$ such that for every hypersurface V defined by a polynomial $f \in \mathcal{C}$, there is a point $\boldsymbol{\alpha} \in \mathcal{H}$ such that $\boldsymbol{\alpha} \notin V$. Taking $\mathcal{F}_{\mathcal{C}}$ to be the family of hypersurfaces defined by polynomials in $\mathcal{C}$, a hitting set is precisely a $\mathcal{F}_{\mathcal{C}}$-evasive 0-subspace family.

The second task generalized by variety-evasive subspace families is the construction of lossless rank condensers. Rank condensers, first appearing in the work of Gabizon and Raz [6], are another of the myriad objects studied in linear-algebraic pseudorandomness. A lossless rank condenser $\mathcal{E} \subseteq \mathbb{F}^{r \times n}$ is a collection of linear maps $\mathbb{F}^n \to \mathbb{F}^r$ such that for every r-dimensional subspace V, there is some map $E \in \mathcal{E}$ where the image of V under E remains r-dimensional. Dualizing, this is precisely the same as requiring the kernel of E to only intersect V at the origin, or equivalently, that the linear spaces in projective space $\mathbb{P}^{n-1}$ corresponding to $\ker E$ and V have empty intersection. Thus, a lossless rank condenser is exactly the same object as an $(n-r-1)$-subspace family in $\mathbb{P}^{n-1}$ that evades all dimension-$(r-1)$ linear spaces.

Besides their generalization of other tasks in algebraic pseudorandomness, explicitly constructing variety-evasive subspace families is a natural task from the point of view of algebraic geometry. For example, as we will see later, variety-evasive subspace families can easily be used to construct Noether normalizing maps for algebraic varieties.

Guo [7] gives the only known construction of variety-evasive subspace families. To describe his result, we need the following function. For $n, d \in \mathbb{N}$ and $k \in \{0, 1, \ldots, n-1\}$, let $N(k, d, n)$ be the function given by

$$N(k,d,n) \coloneqq \min\left(\binom{(k'+1)(n'+1+d)}{(k'+1)d}, \binom{(n-k)(n'+1+d)}{(n-k)d}\right)$$

where $k' \coloneqq \min(k, d-2)$ and $n' \coloneqq n - (k - k')$. Guo [7, Theorem 1.7] constructed (n, d, ε)-evasive k-subspace families of size

$$\operatorname{poly}\left(N(k,d,n), n, \varepsilon^{-1}\right).$$

Moreover, this construction is explicit and can be carried out in time polynomial in the size of the variety-evasive subspace family $\mathcal{H}$ (and $\log p$ if the field $\mathbb{F}$ has characteristic $p > 0$). Guo's work complements this with a lower bound that shows any (n, d)-evasive k-subspace family $\mathcal{H}$ must have size at least

$$|\mathcal{H}| \geq \begin{cases} (n-k)(k+1)+1 & \text{if } d = 1, \\ \max\left(d(n-k)(k+1)+1, \binom{d+n-k}{d} + (n-k+1)k\right) & \text{if } d > 1. \end{cases}$$

This lower bound is tight in the sense that there is a non-explicit construction of a variety-evasive k-subspace family whose size precisely matches the lower bound [7, Section 4].

1.2 Our results

We give an improved construction of (n, d, ε)-evasive k-subspace families. Just like Guo's construction, ours is explicit and can be carried out in time polynomial in the size of the evasive subspace family $\mathcal{H}$ (and $\log p$, if $\mathbb{F}$ has characteristic $p > 0$).

▶ **Theorem 1.4.** *Let $n, d \in \mathbb{N}$, $k \in \{0, 1, \ldots, n-1\}$, and $0 < \varepsilon < 1$. There is an explicit (n, d, ε)-evasive k-subspace family (resp. strongly (n, d, ε)-evasive affine k-subspace family) $\mathcal{H}$ of size $\operatorname{poly}(((n-k)d+1)^{n-k}, \varepsilon^{-1})$. Moreover, there is a deterministic algorithm that on input (n, d, ε) runs in time $\operatorname{poly}(|\mathcal{H}|)$ and prints defining equations of the subspaces in $\mathcal{H}$.*

To compare our construction with that of Guo, we use the simplified bound from [7, Theorem 1.7], which asserts an explicit construction of an (n, d, ε)-evasive k-subspace family of size $\mathrm{poly}(n^{\min(k+1,n-k,d)d}, d, \varepsilon^{-1})$. In contrast, we construct explicit (n, d, ε)-evasive k-subspace families of size $\mathrm{poly}(((n-k)d+1)^{n-k}, \varepsilon^{-1})$. This improves the dependence on d, which may be exponentially larger than n and k; for example, a variety cut out by $n/2$ quadratic equations will generally have degree $2^{n/2}$.

Guo's work shows that in order to construct an (n, d)-evasive k-subspace family (and similarly an (n, d, ε)-evasive k-subspace family), it suffices to construct a hitting set for *Chow forms* of equidimensional projective varieties of dimension $n-k-1$ (see Definition 3.3 for the precise definition). The Chow form of such a variety is a polynomial in $n-k$ sets of $n+1$ variables each, which are thought of as encoding $n-k$ linear equations. The Chow form has the remarkable property that it evaluates to zero exactly when these linear equations have a solution lying on the variety; equivalently, the Chow form is nonzero exactly when the subspace defined by these linear equations evades the variety. If the underlying projective variety has degree d, then the Chow form will be a polynomial of degree d in each set of $n-k$ variables, for a total degree of $(n-k)\cdot d$. Using this bound on the degree, Guo [7] bounds the sparsity of the Chow form and uses the work of Klivans and Spielman [10] to construct a hitting set for Chow forms. To construct (n, d, ε)-evasive subspace families, Guo [7] invokes a result of Agrawal, Gurjar, Korwar, and Saxena [1] that provides a stronger guarantee than a hitting set: they construct an explicit set of points $\mathcal{H}$ such that for all polynomials f of a given sparsity bound, a $1-\varepsilon$ fraction of points in $\mathcal{H}$ hit f. Applying this to Chow forms results in (n, d, ε)-evasive subspace families.

Our Theorem 1.4 stems from an improved construction of hitting sets for Chow forms: we reduce the number of variables from $(n-k)(n+1)$ down to $n-k$ while only increasing the degree by a factor of n. This allows us to construct (n, d)-evasive k-subspace families using hitting sets for polynomials of individual degree dn in $n-k$ variables, resulting in the improved parameters of Theorem 1.4. To obtain (n, d, ε)-evasive k-subspace families, we proceed in the same manner as Guo, applying the work of Agrawal, Gurjar, Korwar, and Saxena [1] to the variable-reduced Chow form.

We also give a simple construction of (n, d)-evasive k-subspace families that eliminates the polynomial overhead appearing in Theorem 1.4 and does not rely on the notion of Chow forms.

▶ **Theorem 1.5.** *Let $n, d \in \mathbb{N}$ and $k \in \{0, 1, \ldots, n-1\}$. There is an explicit (n, d)-evasive k-subspace family (resp. strongly (n, d)-evasive affine k-subspace family) $\mathcal{H}$ of size at most $(nd+1)^{n-k}$. Moreover, there is a deterministic algorithm that receives n, d, and k as input and prints the subspace family $\mathcal{H}$ in time $(nd+1)^{O(n-k)}$.*

When $d \geq n^{1+\varepsilon}$ for some constant $\varepsilon > 0$, Theorem 1.5 is polynomially close to Guo's lower bound on the size of (n, d)-evasive k-subspace families. To see this, observe that we can bound the size of the (n, d)-evasive k-subspace family constructed in Theorem 1.5 by $\left(d^{\frac{2+\varepsilon}{1+\varepsilon}}+1\right)^{(n-k)} \approx d^{\frac{2+\varepsilon}{1+\varepsilon}(n-k)}$. On the other hand, Guo showed that any such k-subspace family must have size at least

$$\binom{d+n-k}{n-k} \geq \left(1+\frac{d}{n-k}\right)^{n-k} \geq \left(1+\frac{d}{n}\right)^{n-k} \geq d^{\frac{\varepsilon}{1+\varepsilon}(n-k)}.$$

As an application of our main result, we give an explicit construction of Noether normalizing maps. Noether normalization is a useful basic result in commutative algebra. In geometric language, the Noether normalization lemma states that for every r-dimensional

affine variety $V \subseteq \mathbb{A}^n$, there is a surjective finite morphism $\pi : V \to \mathbb{A}^r$. Finite morphisms are particularly nice maps of algebraic varieties: every subvariety of V is mapped to a subvariety of $\mathbb{A}^r$, and the preimage $\pi^{-1}(p)$ of any point $p \in \mathbb{A}^r$ will be a finite set.

Over an infinite field, a randomly-chosen linear map $\pi : \mathbb{A}^n \to \mathbb{A}^r$ will be Noether normalizing with high probability. Guo [7, Theorems 1.12 and 1.13] observed that Noether normalizing maps can easily be constructed from an (n, d)-evasive subspace family. As a corollary of Theorem 1.4, we obtain the following explicit construction of Noether normalizing maps.

▶ **Corollary 1.6.** *Let $n, d \in \mathbb{N}$, $r \in \{0, 1, \ldots, n\}$, $k \coloneqq n - r - 1$, and let $\varepsilon > 0$. There is an explicit collection $\mathcal{L}$ of linear maps $\mathbb{A}^n \to \mathbb{A}^r$ of size $\mathrm{poly}(((n-k)d)^{n-k}, \varepsilon^{-1})$ and computable in time $\mathrm{poly}(|\mathcal{L}|)$ such that the following hold.*

1. *For every projective variety $V \subseteq \mathbb{P}^n$ of dimension $r-1$ and degree at most d, at least a $1-\varepsilon$ fraction of maps $\pi \in \mathcal{L}$ induce a surjective finite morphism from V to $\mathbb{P}^{r-1}$.*
2. *For every affine variety $V \subseteq \mathbb{A}^n$ of dimension r and degree at most d, at least a $1-\varepsilon$ fraction of maps $\pi \in \mathcal{L}$ restrict to a surjective finite morphism from V to $\mathbb{P}^r$.*

The proof of Corollary 1.6 is exactly the same as [7, Theorems 1.12 and 1.13], but using the (n, d, ε)-evasive subspace families of Theorem 1.4 instead of the subspace families constructed by Guo, so we omit further details.

2 Preliminaries

Throughout this work, we use $\mathbb{F}$ to denote an algebraically closed field. The ring of polynomials in $n+1$ variables is denoted $\mathbb{F}[x_0, \ldots, x_n]$. We use bold letters for vectors, whose length will be clear from context. We use vector notation to denote monomials: given $\mathbf{a} \in \mathbb{N}^{n+1}$, we write $\mathbf{x}^{\mathbf{a}}$ for the monomial $x_0^{a_0} \cdots x_n^{a_n}$. We use $\mathbb{A}^n$ and $\mathbb{P}^n$ to denote n-dimensional affine and projective spaces, respectively.

2.1 Algebraic geometry

We assume basic algebraic geometry at the level of [13]. For us, an *algebraic variety* is a Zariski-closed subset of $\mathbb{A}^n$ or $\mathbb{P}^n$; in particular, a variety may be reducible. We adopt the convention that the degree of a variety is the sum of the degrees of all of its irreducible components. For this notion of degree, the following inequality holds, which we call Bézout's inequality.

▶ **Lemma 2.1** (Bézout's inequality)**.** *For any two varieties V and W, we have* $\deg(V \cap W) \leq \deg V \cdot \deg W$.

A proof of the affine version of Lemma 2.1 is given in [9, Theorem 1], while a proof of the projective version follows immediately from [5, Chapter 2.3].

We also need the following lemma, which shows that varieties of small degree and dimension are contained in linear subspaces of small dimension.

▶ **Lemma 2.2.** *For every projective variety V, there exists a linear subspace L of dimension at most $(\dim V + 1) \cdot \deg V - 1$ such that $V \subseteq L$.*

Proof. First, suppose V is irreducible. Let $L \subseteq \mathbb{P}^n$ be a minimal linear subspace such that $V \subseteq L$. By [8, Corollary 18.12], we can deduce that V has codimension at most $\deg V - 1$ within L. It follows that L has dimension at most $\dim V + \deg V - 1$.

Now let V be an arbitrary projective variety and let $V_1, \ldots, V_t$ be the irreducible components of V. Each V_i is contained in a linear subspace L_i of dimension at most $\dim V_i + \deg V_i - 1$. Let L be the smallest linear subspace that contains $\bigcup_{i=1}^{t} L_i$. Since each irreducible component V_i is contained in L, the variety V is likewise contained in L. We have

$$\dim L \leq \sum_{i=1}^{t} (\dim L_i + 1) - 1 \leq \sum_{i=1}^{t} (\dim V_i + \deg V_i) - 1.$$

By Bézout's inequality, we have $\sum_{i=1}^{t} \deg V_i \leq \deg V$ and $t \leq \deg V$. Combined with the fact that $\dim V_i \leq \dim V$, the required statement follows. ◀

2.2 Variety-evasive subspace families

In this subsection, we gather some preparatory lemmas for the construction of variety-evasive k-subspace families. In particular, we will see that to construct (n, d, ε)-evasive k-subspace families, it suffices to construct a k-subspace family that evades degree-d varieties of dimension $n - k - 1$, and that we may further assume these varieties live in an ambient space of dimension $(n - k - 1)d$. In addition to this, we will see that an (n, d, ε)-evasive k-subspace family automatically yields an (n, d, ε')-evasive affine k-subspace family with only a mild loss in ε'.

We first cite a lemma of Guo [7] that shows constructing an (n, d, ε)-evasive k-subspace family reduces to the task of constructing a k-subspace that ε-evades equidimensional subvarieties of dimension $n - k - 1$ and degree d.

▶ **Lemma 2.3** ([7, Lemma 3.1]). *Let $\mathcal{F}$ be the family of equidimensional projective subvarieties of $\mathbb{P}^n$ of dimension $n-k-1$ and degree at most d. Suppose $\mathcal{H}$ is an $(\mathcal{F}, \varepsilon)$-evasive k-subspace family. Then $\mathcal{H}$ is (n, d, ε)-evasive.*

Next, we show that the dimension of the ambient space can be reduced. As a consequence of Lemma 2.2, it suffices to construct variety-evasive subspace families when the ambient dimension n is at most $\deg V \cdot \dim V$. Essentially the same reduction is used in the constructions of Guo [7]. We give a proof for completeness.

▶ **Lemma 2.4.** *Let $r := n - k - 1$ and let $\mathcal{H}$ be a given $(rd + d - r - 2)$-subspace family of linear subspaces of $\mathbb{P}^{rd+d-1}$ of size T.*

1. *If $\mathcal{H}$ is $(rd + d - 1, d)$-evasive, then we can construct an (n, d)-evasive k-subspace family of size* $\operatorname{poly}(T, n, d)$ *in time* $\operatorname{poly}(T, n, d)$.
2. *If $\mathcal{H}$ is $(rd + d - 1, d, \varepsilon)$-evasive, then we can construct an $(n, d, 2\varepsilon)$-evasive k-subspace family of size* $\operatorname{poly}\left(T, n, d, \varepsilon^{-1}\right)$ *in time* $\operatorname{poly}\left(T, n, d, \varepsilon^{-1}\right)$.

Proof. We first carry out the construction of an (n, d)-evasive k-subspace family in detail. After this, we briefly remark how to modify the construction to obtain an (n, d, ε)-evasive k-subspace family.

We begin by constructing an $(n, 1)$-evasive $(n - rd - d)$-subspace family $\widehat{\mathcal{H}}$. By Lemma 2.3, it suffices to build a collection $W_1, \ldots, W_s$ of linear subspaces in $\mathbb{P}^n$ such that for any linear subspace $L \subset \mathbb{P}^n$ of dimension $rd + d - 1$, there is some W_i such that $L \cap W_i = \emptyset$. Such a family of subspaces W_i is precisely the nullspaces of the matrices in a *rank extractor*. Forbes and Shpilka [4], together with the improved analysis of Forbes, Saptharishi, and Shpilka [3], gave an explicit construction of such a rank extractor of size $s = (rd + d - 1)(n - rd - d + 1) + 1$. For each $i \in [s]$, we pick points $v_{i,0}, \ldots, v_{i,n} \in \mathbb{P}^n$ such that $\operatorname{span}(v_{i,0}, \ldots, v_{i,n}) = \mathbb{P}^n$ and $\operatorname{span}(v_{i,rd+d}, \ldots, v_{i,n}) = W_i$.

Let $\mathcal{H} = \{Y_1, \ldots, Y_T\} \subseteq \mathbb{P}^{rd}$ be the given $(rd+d-1, d)$-evasive $(rd+d-r-2)$-subspace family. For each Y_i, we pick $rd+d-r-1$ points $u_{i,0}, \ldots, u_{i,rd+d-r-1} \in \mathbb{P}^{rd+d-1}$ that span Y_i.

Now we define linear subspaces $Z_{i,j} \subseteq \mathbb{P}^n$ for every $1 \leq i \leq s$ and $1 \leq j \leq T$ that will form the required (n,d)-evasive k-subspace family. Let $\phi_i : \mathbb{P}^{rd+d-1} \to \mathbb{P}^n$ be the linear map that sends the point e_ℓ to $v_{i,\ell}$, where e_ℓ denotes the point in $\mathbb{P}^{rd+d-1}$ whose ℓ-th coordinate is 1 and whose remaining coordinates are zero. We take $Z_{i,j}$ to be the k-subspace given by

$$Z_{i,j} \coloneqq \operatorname{span}\left(v_{i,rd+d}, \ldots, v_{i,n}, \phi_i(u_{j,0}), \ldots, \phi_i(u_{j,rd+d-r-2})\right).$$

Given the subspace families $\mathcal{H}$ and $\widehat{\mathcal{H}}$, it is clear that we can construct the subspaces $Z_{i,j}$ in $\operatorname{poly}(T, n, d)$ time by iterating over the product set $\mathcal{H} \times \widehat{\mathcal{H}}$.

It remains to show that the k-subspace family $\{Z_{i,j} : i \in [s], j \in [T]\}$ is (n,d)-evasive. To this end, let $V \subseteq \mathbb{P}^n$ be an equidimensional variety of dimension r and degree d. By Lemma 2.2, there is a linear subspace X of dimension $rd+d-1$ that contains V. Since $\widehat{\mathcal{H}} = \{W_1, \ldots, W_s\}$ is $(n,1)$-evasive, there is some W_i such that $W_i \cap X = \varnothing$; without loss of generality, we may assume that $W_1 \cap X = \emptyset$. For each $0 \leq i \leq rd+d-1$, the intersection $\operatorname{span}(W_1, v_{i,1}) \cap X$ consists of a single point in $\mathbb{P}^n$, which we denote by a_i. Comparing dimensions, we see that $\operatorname{span}(a_0, \ldots, a_{rd+d-1}) = X$. Let $\psi : \mathbb{P}^{rd+d-1} \to \mathbb{P}^n$ denote the linear map that maps e_i to a_i. Since $v_{1,i} \in \operatorname{span}(W_1, \psi(e_i))$, it follows that $\operatorname{span}(W_1, \psi(e_i)) = \operatorname{span}(W_1, \phi_1(e_i)) = \operatorname{span}(W_1, v_{i,1})$.

The image of the map ψ is exactly X, and so ψ is an isomorphism between $\mathbb{P}^{rd+d-1}$ and X. Since $V \subseteq X$, the preimage of V under ψ is also a variety of dimension r and degree d. Because $\mathcal{H} = \{Y_1, \ldots, Y_T\}$ is $(rd+d-1, d)$-evasive, there is some Y_j such that $Y_j \cap \psi^{-1}(V) = \varnothing$; without loss of generality, we may assume that Y_1 does not intersect $\psi^{-1}(V)$. The choice of the map ψ implies that $Z_{1,1} = \operatorname{span}(W_1, \psi(Y_1))$. Now suppose towards a contradiction that there is a point $b \in V \cap Z_{1,1}$. The point b lies in $\operatorname{span}(\psi(c), w)$, for some $c \in Y_1$ and $w \in W_1$. Since $b, \psi(c) \in X$ and $X \cap W_1 = \emptyset$, we deduce $b = \psi(c)$ in $\mathbb{P}^n$. This contradicts the fact that $Y_1 \cap \psi^{-1}(V) = \emptyset$.

Since V was an arbitrary equidimensional variety, we can invoke Lemma 2.3 to conclude that the constructed k-subspace is (n,d)-evasive. To construct an (n,d,ε)-evasive k-subspace family, the only part of the construction that changes is that the $(n,1)$-evasive subspace family used in the first step must be replaced by an $(n,1,\varepsilon)$-evasive subspace family. The Forbes–Shpilka construction of rank extractors also produces this $(n,1,\varepsilon)$-evasive subspace family, incurring only a $1/\varepsilon$ multiplicative increase in the size of the subspace family. ◀

Finally, we show that any (n,d,ε)-evasive k-subspace family immediately yields an (n,d,ε')-evasive affine k-subspace family with only a small loss in the parameter ε'.

▶ **Lemma 2.5.** *Let $\mathcal{H}$ be an (n,d,ε)-evasive k-subspace family. Let $\mathcal{H}' \subseteq \mathcal{H}$ be the subfamily of subspaces not contained in the hyperplane at infinity. Then the restriction of $\mathcal{H}'$ to the affine chart of points at finite distance is a strongly $(n,d,2\varepsilon/(1-\varepsilon))$-evasive affine k-subspace family.*

Proof. Since $\mathcal{H}$ is (n,d,ε)-evasive, at most ε fraction of the elements of $\mathcal{H}$ are contained in $\mathrm{V}(x_0)$, the hyperplane at infinity, so $|\mathcal{H}'| \geq (1-\varepsilon)|\mathcal{H}|$. For the rest of this proof, let $\varepsilon' \coloneqq \varepsilon/(1-\varepsilon)$. The family $\mathcal{H}'$ is (n,d,ε')-evasive.

Let $\mathcal{H}_\infty$ be the restriction to $\mathrm{V}(x_0)$ of those subspaces in $\mathcal{H}'$. This is a $(k-1)$-subspace family of size at least $(1-\varepsilon)|\mathcal{H}|$. Let $V \subseteq \mathbb{P}^{n-1}$ be any variety of degree at most d. We can identify $\mathbb{P}^{n-1}$ with the hyperplane at infinity in $\mathbb{P}^n$. This way, V can be viewed as a variety in $\mathbb{P}^n$. All except $\varepsilon|\mathcal{H}|$ elements in $\mathcal{H}'$ evade V. Therefore, $\mathcal{H}_\infty$ is a $(n-1,d,\varepsilon')$-evasive subspace family.

We now show that the restriction of $\mathcal{H}'$ to the affine chart is strongly $(n, d, 2\varepsilon')$-evasive for affine varieties. Let $V \subseteq \mathbb{A}^n$ be an affine variety of degree d. Identifying $\mathbb{A}^n$ with the affine chart, we can consider $V \subseteq \mathbb{P}^n$. Let $\overline{V}$ be the projective closure of V, and let V_∞ denote $\overline{V} \cap \mathrm{V}(x_0)$. Both $\overline{V}$ and V_∞ have degree at most d.

Suppose $H \in \mathcal{H}'$ is such that H evades $\overline{V}$ and $H \cap \mathrm{V}(x_0)$ evades V_∞. Let W be an irreducible component of V, let $\overline{W}$ be the projective closure of W, and let W_∞ denote $\overline{W} \cap \mathrm{V}(x_0)$. By assumption, every component of $\overline{W} \cap H$ has dimension $\dim \overline{W} + k - n$. If $\dim \overline{W} + k - n < 0$, then this intersection is empty. In this case, $W \cap H \cap \mathbb{A}^n$ is also empty, and therefore $H \cap \mathbb{A}^n$ evades W. Otherwise, we have $\dim \overline{W} + k - n \geq 0$.

To show that $H \cap \mathbb{A}^n$ evades W, it suffices to show that $\overline{W} \cap H \not\subseteq \mathrm{V}(x_0)$. This will guarantee that $W \cap H \cap \mathbb{A}^n$ is nonempty and has the correct dimension. To this end, note that $\dim W_\infty = \dim W - 1$. Since $H \cap \mathrm{V}(x_0)$ evades W_∞, every component of $\overline{W} \cap H \cap \mathrm{V}(x_0)$ has dimension $\dim \overline{W} + k - n - 1$. Since every component of $\overline{W} \cap H$ has dimension $\dim \overline{W} + k - n$, no component of $\overline{W} \cap H$ lies completely in the hyperplane at infinity, completing the proof that $H \cap \mathbb{A}^n$ evades W. Since this holds for every component W of V, we deduce that $H \cap \mathbb{A}^n$ evades V.

Because $\mathcal{H}'$ and $\mathcal{H}_\infty$ are (n, d, ε')-evasive and $(n-1, d, \varepsilon')$-evasive, respectively, the fraction of hyperplanes in $\mathcal{H}'$ such that either H does not evade $\overline{V}$ or such that $H \cap \mathrm{V}(x_0)$ does not evade V_∞ is at most $2\varepsilon'$. Therefore, the restriction of $\mathcal{H}'$ to the affine chart is a strongly $(n, d, 2\varepsilon')$-evasive affine k-subspace family. ◀

3 Construction of variety-evasive subspace families

We give two constructions of variety-evasive subspace families. The first is a simple construction meant to illustrate our main technical observation. The second is a more involved construction that allows us to obtain improved epsilon variety-evasive subspace families.

3.1 Basic construction

Let $V \subseteq \mathbb{P}^n$ be an projective variety of dimension r. The intersection of V with a general hyperplane is a variety of dimension $r - 1$. Typically, to obtain an effective version of this fact, one picks a point on each irreducible component of V and controls the probability that a randomly-chosen hyperplane H does not pass through the chosen points. If H does not pass through these points, then H properly intersects each component of V, so the intersection $V \cap H$ will have dimension exactly $r - 1$.

We implement this argument in the following lemma. Here, the coefficients of the hyperplane H are not picked independently, but are instead chosen from points of the form $(1, \gamma, \gamma^2, \ldots, \gamma^n)$ for $\gamma \in \mathbb{F}$. When we invoke this lemma in the proof of Theorem 1.5, we will need to choose a hyperplane H that properly intersects a collection of irreducible varieties that need not be the irreducible components of a single variety. For example, one variety in the given collection may be properly contained in another.

▶ **Lemma 3.1.** *Let $V_1, \ldots, V_t \subseteq \mathbb{P}^n$ be irreducible projective varieties. There exists a polynomial $P \in \mathbb{F}[z]$ of degree nt such that for any γ with $P(\gamma) \neq 0$, every hyperplane section $V_i \cap \mathrm{V}\left(\sum_{j=0}^n \gamma^j x_j\right)$ has dimension $\dim V_i - 1$.*

Proof. From each variety V_i of V, we pick a point p_i. Consider the polynomial $Q(\mathbf{x}, z) \coloneqq \sum_{i=0}^n z^i x_i$. The evaluation $Q(p_i, z)$ is a nonzero polynomial in z, since the point p_i has at least one nonzero coordinate. Set $P(z) \coloneqq \prod_{i=1}^t Q(p_i, z)$.

It is clear that P is a polynomial of degree nt. To show that P satisfies the claimed property, let $\gamma \in \mathbb{F}$ be a point such that $P(\gamma) \neq 0$ and consider the linear form $L(\mathbf{x}) \coloneqq \sum_{i=0}^{n} \gamma^i x_i$. By construction, we have that $L(p_i) = Q(p_i, \gamma)$. Because $P(\gamma) \neq 0$, it follows that $Q(p_i, \gamma) \neq 0$. In particular, the linear form L does not vanish identically on V_i. Therefore $V_i \cap \mathrm{V}(L)$ has dimension $\dim V_i - 1$ by [2, Chapter 9, Section 4, Corollary 4]. ◀

We can also deduce a similar statement for affine varieties.

▶ **Lemma 3.2.** *Let $V_1, \ldots, V_t \subseteq \mathbb{A}^n$ irreducible affine varieties of degree $d_1, \ldots, d_t$, respectively. There exists a polynomial $P \in \mathbb{F}[z]$ of degree at most $n \sum_{i=1}^{t} d_i$ such that for any γ with $P(\gamma) \neq 0$, every hyperplane section $V_i \cap \mathrm{V}\left(1 + \sum_{j=1}^{n} \gamma^j x_j\right)$ has dimension $\dim V_i - 1$.*

Proof. Let $\overline{V}_i$ denote the projective closure of the variety V_i, which is an irreducible projective variety. Because the variety $\overline{V}_i$ is irreducible, we have $\dim(\overline{V}_i \cap \mathrm{V}(x_0)) = \dim \overline{V}_i - 1$. For each i such that $\dim \overline{V}_i \geq 1$, and for each irreducible component $\overline{V}_{i,j}$ of $\overline{V}_i \cap \mathrm{V}(x_0)$, pick a point $p_{i,j} \in \overline{V}_{i,j}$. For those i such that $\dim \overline{V}_i = 0$, let $p_{i,1}$ be the single point of $\overline{V}_i$.

Consider the polynomial $Q(\mathbf{x}, z) \coloneqq \sum_{i=0}^{n} z^i x_i$. If we evaluate at $p_{i,j}$, we get $Q(p_{i,j}, z)$, which is a nonzero polynomial in $\mathbb{F}[z]$, since $p_{i,j}$ has at least one nonzero coordinate. Set $P(z) \coloneqq \prod_{i,j} Q(p_{i,j}, z)$. The fact that $P(z)$ has degree at most $n \sum_{i=1}^{t} d_i$ follows from the fact that each factor $Q(p_{i,j}, z)$ is a polynomial of degree n, together with the fact that there is one factor of $P(z)$ per component of $\overline{V}_i$, of which there are at most $d_i = \deg V_i$. If we pick $\gamma \in \mathbb{F}$ such that $P(\gamma) \neq 0$, then the linear form $L(\mathbf{x}) \coloneqq \sum_{i=0}^{n} \gamma^i x_i$ is nonzero at every $p_{i,j}$. From this, we can deduce that $\dim(\overline{V}_i \cap \mathrm{V}(L, x_0)) = \dim \overline{V}_i - 2$ for every variety $\overline{V}_i$ of dimension at least one. Therefore, for any such variety, we must have $\dim(V_i \cap \mathrm{V}(\ell)) = \dim V_i - 1$, where $\ell(\mathbf{x}) \coloneqq L(1, x_1, \ldots, x_n)$. For the zero-dimensional varieties V_i, we can also deduce that $V_i \cap \mathrm{V}(\ell) = \emptyset$, since L does not vanish at the single point of V_i. ◀

Applying Lemma 3.1 to the irreducible components of a projective variety allows us to construct a set $\mathcal{H}$ of $nd + 1$ hyperplanes such that for any projective variety $V \subseteq \mathbb{P}^n$ of degree d, there is some $H \in \mathcal{H}$ that properly intersects V. Lemma 3.2 does the same for affine varieties. By iterating this construction k times, we can obtain an (n, d)-evasive (affine) k-subspace family. We remark that this solves [7, Open Problem 7], which asked for an explicit construction of a polynomial-size (n, d)-evasive k-subspace family when $n - k = O(1)$.

▶ **Theorem 1.5.** *Let $n, d \in \mathbb{N}$ and $k \in \{0, 1, \ldots, n-1\}$. There is an explicit (n, d)-evasive k-subspace family (resp. strongly (n, d)-evasive affine k-subspace family) $\mathcal{H}$ of size at most $(nd+1)^{n-k}$. Moreover, there is a deterministic algorithm that receives n, d, and k as input and prints the subspace family $\mathcal{H}$ in time $(nd+1)^{O(n-k)}$.*

Proof. We start by establishing the projective version of the theorem. Fix pairwise disjoint subsets $B_1, \ldots, B_{n-k} \subseteq \mathbb{F}$, each of size $nd+1$ and not containing 0. For each $\alpha \in \mathbb{F}$, let $L_\alpha(\mathbf{x})$ denote the linear form $x_0 + \alpha x_1 + \cdots + \alpha^n x_n$. For each $\boldsymbol{\gamma} \in \mathbb{F}^{n-k}$, let $H_{\boldsymbol{\gamma}} \subseteq \mathbb{P}^n$ denote the linear subspace $\mathrm{V}\left(L_{\gamma_1}, \ldots, L_{\gamma_{n-k}}\right)$. Let $\mathcal{H}$ denote the collection of subspaces $H_{\boldsymbol{\gamma}}$ where $\boldsymbol{\gamma}$ varies over elements in $B_1 \times \cdots \times B_{n-k}$. Finally, let $\mathcal{H}'$ denote the restriction of each element of $\mathcal{H}$ to the affine chart $x_0 = 1$. We claim that $\mathcal{H}$ and $\mathcal{H}'$ are the required (n, d)-evasive k-subspace and affine k-subspace families, respectively. The bound on the sizes of $\mathcal{H}$ and $\mathcal{H}'$ are clear from construction. To algorithmically produce $\mathcal{H}$ and $\mathcal{H}'$, we simply iterate over the set $B_1 \times \cdots \times B_{n-k}$ and print the subspace $H_{\boldsymbol{\gamma}}$ for every $\boldsymbol{\gamma} \in B_1 \times \cdots \times B_{n-k}$.

To see why $\mathcal{H}$ is (n, d)-evasive, let $V \subseteq \mathbb{P}^n$ be any variety of degree d. By applying Lemma 3.1 to the irreducible components $V_1 \cup \cdots \cup V_t$ of V, there exists a polynomial $P_1 \in \mathbb{F}[z]$ of degree $nt \leq nd$ such that for any $\gamma_1 \in \mathbb{F}$ satisfying $P_1(\gamma_1) \neq 0$, the intersection of V_i and $\mathrm{V}(L_{\gamma_1})$ has dimension $\dim V_i - 1$ for every irreducible component V_i of V. Since the set B_1 has size $nd + 1$, we can find such an element $\gamma_1 \in B_1$. Fix this choice of γ_1.

Let d_i be the degree of the component V_i. By Bézout's inequality (Lemma 2.1), the intersection $V_i \cap \mathrm{V}(L_{\gamma_1})$ has degree at most d_i, and thus consists of at most d_i irreducible components. We again apply Lemma 3.1 to the collection of varieties consisting of the irreducible components of $V_1 \cap \mathrm{V}(L_{\gamma_1}), \ldots, V_t \cap \mathrm{V}(L_{\gamma_1})$, of which there are at most $d_1 + \cdots + d_t = d$. This gives us a polynomial P_2 of degree at most nd whose non-roots define hyperplanes that properly intersect each of these components. Because $|B_2| = nd + 1$, we can find a nonroot γ_2 of P_2 that lies in B_2. Moreover, since B_1 and B_2 are disjoint, we are guaranteed that γ_2 is different from γ_1. This implies that the intersection $V_i \cap \mathrm{V}(L_{\gamma_1}) \cap \mathrm{V}(L_{\gamma_2})$ has dimension $\dim V_i - 2$ for every irreducible component V_i of V. Continuing in this manner, we obtain $(\gamma_1, \ldots, \gamma_{n-k}) \in B_1 \times \cdots \times B_{n-k}$ such that the subspace $H_{\boldsymbol{\gamma}}$ evades V.

The affine version for $\mathcal{H}'$ follows the same argument, invoking Lemma 3.2 in place of Lemma 3.1. The fact that no B_i contains 0 implies that each affine subspace in $\mathcal{H}'$ has dimension k. ◀

3.2 Constructing (n, d, ε)-evasive subspace families

Theorem 1.5 constructs an (n, d)-evasive k-subspace family by building the subspaces iteratively, intersecting $n - k$ hyperplanes one at a time. To improve this to a construction of an (n, d, ε)-evasive k-subspace family, one natural approach is to ensure that at each step, the probability of picking a hyperplane that does not properly intersect a variety V is bounded by $\frac{\varepsilon}{n-k}$. Applying a union bound over the $n - k$ choices of hyperplanes implies that at most an ε fraction of subspaces fail to evade V. However, this enlarges the number of hyperplanes chosen at each step by a factor of $\frac{n-k}{\varepsilon}$, which leads to a $(1/\varepsilon)^{n-k}$ factor in the final size of the subspace family.

By arguing about the $n - k$ hyperplanes as a whole instead of one at a time, we can improve this $(1/\varepsilon)^{n-k}$ factor to $(1/\varepsilon)^{O(1)}$. To do this requires the notion of the Chow form of a variety, which we now define. The definition and facts we use about the Chow form are classical. The following statements appear in [11, Section 2.1.1], and proofs can be found in [12, Chapter 1].

▶ **Definition 3.3.** *Let $V \subseteq \mathbb{P}^n$ be an equidimensional variety of dimension r and degree d. Let $\mathbf{u}_0, \ldots, \mathbf{u}_r$ be sets of variables, each of size $n+1$. The* Chow form of V, *denoted $\mathcal{C}_V$, is a polynomial in $\mathbb{F}[\mathbf{u}_0, \ldots, \mathbf{u}_r]$ with the following properties.*

- *For every choice of $\boldsymbol{\gamma}_0, \ldots, \boldsymbol{\gamma}_r \in \mathbb{F}^{n+1}$, the linear subspace $\mathrm{V}\left(\sum_{j=0}^n \gamma_{0,j} x_j, \ldots, \sum_{j=0}^n \gamma_{r,j} x_j\right)$ has a point in V if and only if $\mathcal{C}_V(\boldsymbol{\gamma}_0, \ldots, \boldsymbol{\gamma}_r) = 0$.*
- *The polynomial $\mathcal{C}_V$ squarefree and homogeneous of degree d in each set of variables $\mathbf{u}_i$.*

In other words, the Chow form of a variety characterizes those linear subspaces of dimension at least $n - r + 1$ that intersect V. The following is an easy corollary of Lemma 3.1.

▶ **Corollary 3.4.** *Let $V \subseteq \mathbb{P}^n$ be an equidimensional variety of dimension r and degree d with Chow form $\mathcal{C}_V \in \mathbb{F}[\mathbf{u}_0, \ldots, \mathbf{u}_r]$. Let $v_0, \ldots, v_r$ be new variables and let $\phi : \mathbb{F}[\mathbf{u}_0, \ldots, \mathbf{u}_r] \to \mathbb{F}[\mathbf{v}]$ be the map defined by $\phi(u_{i,j}) = v_i^j$. Then $\phi(\mathcal{C}_V) \neq 0$.*

Proof. Let $\gamma_0 \in \mathbb{F}$ be such that $V_1 \coloneqq V \cap \mathrm{V}(x_0 + \gamma_0 x_1 + \cdots + \gamma_0^n x_n)$ is equidimensional of dimension $r - 1$. Such an element exists by Lemma 3.1. The variety V_1 is now equidimensional of dimension $r - 1$, and we can invoke Lemma 3.1 again to find a $\gamma_1 \in \mathbb{F}$ such that $V_2 \coloneqq V_1 \cap \mathrm{V}(x_0 + \gamma_1 x_1 + \cdots + \gamma_1^n x_n)$ is equidimensional of dimension $r - 2$. Repeating this, we find $\gamma_0, \ldots, \gamma_r$ such that $V \cap \mathrm{V}\left(\sum_{j=0}^n \gamma_0^j x_j, \ldots, \sum_{j=0}^n \gamma_r^j x_j\right)$ is empty. By Definition 3.3, we have $\phi(\mathcal{C}_V)(\gamma_0, \ldots, \gamma_r) \neq 0$, which shows in particular that $\phi(\mathcal{C}_V)$ is nonzero. ◀

In the notation of Corollary 3.4, the polynomial $\phi(\mathcal{C}_V)$ is a nonzero polynomial of individual degree nd in $r+1$ variables with the property that the subspace defined by the linear forms with coefficients $\boldsymbol{\gamma}_0, \ldots, \boldsymbol{\gamma}_r$ evades V if $\phi(\mathcal{C}_V)(\boldsymbol{\gamma}_0, \ldots, \boldsymbol{\gamma}_r) \neq 0$. The construction of Theorem 1.5 can be thought of as a construction of a hitting set for the restricted Chow form $\phi(\mathcal{C}_V)$. To obtain (n, d, ε)-evasive k-subspace families, we instead apply an ε-hitting set to the restricted Chow form $\phi(\mathcal{C}_V)$. An *ε-hitting set* has exactly the guarantee we need: it is a finite set $\mathcal{H}$ such that a $1-\varepsilon$ fraction of points $(\boldsymbol{\gamma}_0, \ldots, \boldsymbol{\gamma}_r) \in \mathcal{H}$ satisfy $\phi(\mathcal{C}_V)(\boldsymbol{\gamma}_0, \ldots, \boldsymbol{\gamma}_r) \neq 0$. We make this observation precise in the following corollary.

▶ **Corollary 3.5.** *Suppose $\mathcal{P}$ is an ε-hitting set for polynomials of individual degree nd in $n-k$ variables. Then there exists a (n, d, ε)-evasive k-subspace family $\mathcal{H}$ of size at most $|\mathcal{P}|$. Moreover, there is a deterministic algorithm that takes $\mathcal{P}$ as input, prints the set $\mathcal{H}$, and runs in time* $\mathrm{poly}(n, |\mathcal{P}|)$.

Proof. The evasive family of subspaces is constructed as follows. For each point $\boldsymbol{\mu} \in \mathcal{P}$, let $H_{\boldsymbol{\mu}}$ be the linear subspace defined by the linear forms $x_0 + \mu_i x_1 + \cdots + \mu_i^n x_n$ for $i \in [n-k]$. Let $\mathcal{H}'$ be the resulting family of subspaces.

Let V be an arbitrary equidimensional variety of dimension $n-k-1$ and degree d, and let $\mathcal{C}_V$ be its Chow form. Let ϕ be the map defined in Corollary 3.4, so $\phi(\mathcal{C}_V)$ is a nonzero polynomial of individual degree nd in $n-k$ variables. Since $\mathcal{P}$ is a ε-hitting set for such polynomials, for all except $\varepsilon|\mathcal{P}|$ elements in $\mathcal{P}$, we have $\phi(\mathcal{C}_V)(\boldsymbol{\mu}) \neq 0$. Therefore, for all except $\varepsilon|\mathcal{H}'|$ of the subspaces $H \in \mathcal{H}'$, we have $V \cap H = \emptyset$.

Note that the exceptional set of $\varepsilon|\mathcal{H}'|$ vector spaces $H \in \mathcal{H}'$ such that $V \cap H \neq \emptyset$ contains all those subspaces in $\mathcal{H}'$ that are not k-dimensional. Therefore, if we define $\mathcal{H}$ to be the set of k-dimensional subspaces in $\mathcal{H}'$, then all except $\varepsilon|\mathcal{H}|$ elements in $\mathcal{H}$ evade V. Since V was an arbitrary equidimensional variety of dimension $n-k-1$ and degree d, we can invoke Lemma 2.3 to deduce that $\mathcal{H}$ is (n, d, ε)-evasive.

It is clear that the set $\mathcal{H}'$ can be obtained in polynomial time from the ε-hitting set $\mathcal{P}$. To prune $\mathcal{H}'$ to the (n, d, ε)-evasive k-subspace family $\mathcal{H}$, we need to remove from $\mathcal{H}'$ those subspaces that are not k-dimensional. This is a basic linear algebra calculation that can be performed in $\mathrm{poly}(n)$ time for each element of $\mathcal{H}'$. ◀

Finally, we need an explicit construction of ε-hitting sets. We do this using a construction of Agrawal, Gurjar, Korwar, and Saxena [1], as instantiated by Guo [7]. The version we state here is adapted to polynomials of bounded individual degree instead of bounded total degree, but follows exactly the same proof as [7, Lemma 2.3].

▶ **Lemma 3.6** ([7, Lemma 2.3], [1, Lemma 4])**.** *For every $d, n \in \mathbb{N}$ and $\varepsilon \in (0,1)$, there exists an ε-hitting set $\mathcal{P}$ for the set of all polynomials of individual degree d in n variables. The set $\mathcal{P}$ has size and bit complexity* $\mathrm{poly}\left((d+1)^n, \varepsilon^{-1}\right)$ *and can be constructed in time* $\mathrm{poly}(|\mathcal{P}|)$.

Combining Corollary 3.5, Lemma 3.6, and the reductions in Section 2, we obtain our final construction of (n, d, ε)-evasive subspaces.

▶ **Theorem 1.4.** *Let $n, d \in \mathbb{N}$, $k \in \{0, 1, \ldots, n-1\}$, and $0 < \varepsilon < 1$. There is an explicit (n, d, ε)-evasive k-subspace family (resp. strongly (n, d, ε)-evasive affine k-subspace family) $\mathcal{H}$ of size* $\mathrm{poly}(((n-k)d+1)^{n-k}, \varepsilon^{-1})$. *Moreover, there is a deterministic algorithm that on input (n, d, ε) runs in time* $\mathrm{poly}(|\mathcal{H}|)$ *and prints defining equations of the subspaces in $\mathcal{H}$.*

Proof. We start by constructing an (n, d, ε)-evasive k-subspace family. First suppose $n \leq (n-k)d$. Then the required (n, d, ε)-evasive k-subspace family can be constructed by using Corollary 3.5 instantiated with the ε-hitting set from Lemma 3.6. The resulting variety-evasive subspace family has size and bit complexity $\mathrm{poly}\left((nd+1)^{n-k}, \varepsilon^{-1}\right) \leq \mathrm{poly}\left(((n-k)d+1)^{n-k}, \varepsilon^{-1}\right)$, since $n \leq (n-k)d$.

Now suppose $n > (n-k)d$. Let $r := n-k-1$. Using Lemma 3.6 and Corollary 3.5, we can construct an $(rd, d, \varepsilon/2)$-evasive $(rd-r-1)$-subspace family. This family has size $\text{poly}\left((rd+1)^r, \varepsilon^{-1}\right) \leq \text{poly}\left(((n-k)d+1)^{n-k}, \varepsilon^{-1}\right)$. We can now invoke Lemma 2.4 to obtain the required (n, d, ε)-evasive subspace family. The size of the resulting family is also bounded by $\text{poly}\left(((n-k)d+1)^{n-k}, \varepsilon^{-1}\right)$.

To construct a strongly (n, d, ε)-evasive affine k-subspace family, we first pick ε' such that $\varepsilon = 2\varepsilon'/(1-\varepsilon')$. We construct a (n, d, ε')-evasive k-subspace family using the above construction and then invoke Lemma 2.5 to obtain the required (n, d, ε)-evasive affine k-subspace family. ◀

References

1. Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *SIAM J. Comput.*, 44(3):669–697, 2015. doi:10.1137/140975103.
2. David A. Cox, John Little, and Donal O'Shea. *Ideals, varieties, and algorithms - an introduction to computational algebraic geometry and commutative algebra*. Undergraduate texts in mathematics. Springer, 4 edition, 2015. doi:10.1007/978-3-319-16721-3.
3. Michael A. Forbes, Ramprasad Saptharishi, and Amir Shpilka. Hitting sets for multilinear read-once algebraic branching programs, in any order. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC 2014)*, pages 867–875, 2014. doi:10.1145/2591796.2591816.
4. Michael A. Forbes and Amir Shpilka. On identity testing of tensors, low-rank recovery and compressed sensing. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC 2012)*, pages 163–172, 2012. doi:10.1145/2213977.2213995.
5. William Fulton. *Introduction to intersection theory in algebraic geometry*, volume 54. American Mathematical Soc., 1984.
6. Ariel Gabizon and Ran Raz. Deterministic extractors for affine sources over large fields. *Combinatorica*, 28:415–440, 2008. doi:10.1007/s00493-008-2259-3.
7. Zeyu Guo. Variety evasive subspace families. *computational complexity*, 33(2):10, 2024. doi:10.1007/s00037-024-00256-1.
8. Joe Harris. *Algebraic geometry: a first course*, volume 133. Springer Science & Business Media, 2013. doi:10.1007/978-1-4757-2189-8.
9. Joos Heintz. Definability and fast quantifier elimination in algebraically closed fields. *Theor. Comput. Sci.*, 24(3):239–277, 1983. doi:10.1016/0304-3975(83)90002-6.
10. Adam R. Klivans and Daniel Spielman. Randomness efficient identity testing of multivariate polynomials. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC 2001)*, STOC '01, pages 216–223. Association for Computing Machinery, 2001. doi:10.1145/380752.380801.
11. Teresa Krick, Luis Miguel Pardo, and Martín Sombra. Sharp estimates for the arithmetic Nullstellensatz. *Duke Mathematical Journal*, 109(3):521–598, 2001. doi:10.1215/S0012-7094-01-10934-4.
12. Patrice Philippon. Critères pour l'indépendance algébrique. *Inst. Hautes Études Sci. Publ. Math.*, pages 5–52, 1986. doi:10.1007/BF02699191.
13. Igor R. Shafarevich. *Basic Algebraic Geometry 1*. Springer Berlin, Heidelberg, second edition, 1994. doi:10.1007/978-3-642-57908-0.
14. Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.

Testing Unate Distributions

Daeho Lee
Massachusetts Institute of Technology, Cambridge, MA, USA

Shivam Nadimpalli
Massachusetts Institute of Technology, Cambridge, MA, USA

Mingda Qiao
University of Massachusetts Amherst, MA, USA

Ronitt Rubinfeld
Massachusetts Institute of Technology, Cambridge, MA, USA

Abstract

We initiate the study of *unate distributions* over $\{\pm 1\}^n$ – a natural analogue of unate Boolean functions – by considering two basic testing problems that parallel well-studied questions for monotone distributions:

- **Uniformity Testing of Unate Distributions:** We show that $\widetilde{\Theta}(n^{3/2})$ samples are sufficient and necessary, in contrast to the $\widetilde{\Theta}(n)$ sample complexity of the analogous problem for monotone distributions (Rubinfeld and Servedio, STOC 2005; Adamaszek, Czumaj, and Sohler, SODA 2010).
- **Unateness Testing of Arbitrary Distributions:** We give a tester that uses $\widetilde{O}(n^{3/2})$ conditional samples in the subcube conditional model. On the other hand, every tester that draws conditional samples in a similar fashion, namely from $O(1)$-dimensional subcubes, must have an $\widetilde{\Omega}(n^{2/3})$ complexity. In the same model, the complexity of monotonicity testing was recently shown to be $\widetilde{\Theta}(n)$ (Chakrabarty et al., STOC 2025).

Our algorithms for both problems significantly outperform the naive approach of reducing to the monotone case, which would incur $\Omega(n^2)$ sample complexity. Our uniformity tester relies on a subroutine that "weakly" learns the hidden orientations of a unate distribution, together with a new correlation bound for these estimates. Both tools may be of independent interest in studying monotonicity and unateness over $\{\pm 1\}^n$.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms; Mathematics of computing → Probability and statistics

Keywords and phrases Distribution testing, unate distributions, monotone distributions, uniformity testing, subcube conditioning

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.57

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2607.01573` [45]

Funding *Daeho Lee*: Supported by the MIT Undergraduate Research Opportunities Program (UROP).
Shivam Nadimpalli: Partially supported by Elchanan Mossel's Vannevar Bush Faculty Fellowship ONR-N00014-20-1-2826.
Ronitt Rubinfeld: Supported by the NSF TRIPODS program (award DMS-2022448) and CCF-2310818.



© Daeho Lee, Shivam Nadimpalli, Mingda Qiao, and Ronitt Rubinfeld;
licensed under Creative Commons License CC-BY 4.0



1 Introduction

Many fundamental distribution testing tasks – uniformity testing being the canonical example – become infeasible in high dimensional settings such as the Boolean hypercube $\{\pm 1\}^n$, requiring exponentially many samples in the dimension [15, 18]. A central theme in the field has therefore been to identify *structured* settings that admit efficient testers: examples include Bayesian networks [17, 35], Markov random fields [34, 8], and structured truncations [36, 37].

Among such structural assumptions, one of the most extensively studied is *monotonicity*, the distributional analogue of the classical Boolean function property.[1] In the Boolean function setting, monotonicity has played a central role in sublinear algorithms for nearly three decades: it was among the first properties studied in Boolean function testing and continues to be actively investigated to this day [14, 38, 22, 29, 41, 43, 26, 11, 44, 25]. In the distributional setting, monotonicity has similarly been studied in depth, leading to tight upper and lower bounds for tasks such as uniformity testing and property testing under various access models [6, 51, 2, 9, 1, 4, 52, 12, 21].

A closely related property in the Boolean function setting is *unateness*. Informally, a Boolean function is unate if each coordinate is either always non-decreasing or always non-increasing. The problem of testing unateness of Boolean functions was introduced alongside monotonicity testing in [38], and has since received considerable attention [42, 23, 31, 32, 30, 5]. Both monotonicity and unateness are fundamental structural properties that arise naturally across many domains, for example in social choice (where we can view a Boolean function f as a voting rule), economics, learning theory, and circuit design.

Despite extensive work on unate functions, the analogous notion for distributions has, to the best of our knowledge, not been studied. In this work, we initiate the study of *unate distributions* over $\{\pm 1\}^n$. The motivation for this is twofold:

- The study of monotone distributions has a well-developed literature precisely because monotonicity is a natural structural constraint arising in settings ranging from social choice to learning theory. Unateness is a strict and natural generalization – it captures the same "directional" structure without fixing a canonical orientation – and we believe its study for distributions will prove equally fruitful.
- Additionally, unate distributions, which we define shortly, form a broad and natural family of probability distributions. For example, every unate function f induces a unate distribution (the uniform distribution over $f^{-1}(1)$), and the class of unate functions includes families such as halfspaces and read-once decision lists, which need not be monotone. Beyond this connection to functions, every product distribution over $\{\pm 1\}^n$ is unate.

Finally, as this work demonstrates, unate distributions pose challenges that cannot be addressed by simply reducing to the monotone setting. Even for basic tasks such as uniformity testing, the sample complexity changes, and more broadly, new ideas are required to design and analyze optimal testers for this richer class of distributions.

1.1 Our Results

We start by formally defining unate distributions over $\{\pm 1\}^n$:

▶ **Definition 1.** *A distribution $\mathcal{D}$ over $\{\pm 1\}^n$ is* unate *if there exists $\sigma \in \{\pm 1\}^n$ such that $\mathcal{D}(x \oplus \sigma)$ is a monotone probability mass function, where $\oplus$ denotes coordinate-wise multiplication.*

[1] Formally, a distribution $\mathcal{D}$ over $\{\pm 1\}^n$ is *monotone* if its probability mass function $\mathcal{D} : \{\pm 1\}^n \to [0, 1]$ is monotone, i.e., $\mathcal{D}(x) \leq \mathcal{D}(y)$ whenever $x_i \leq y_i$ for all $i \in [n]$.

Unate distributions immediately inherit many of the algorithmic challenges and lower bounds known from the monotone distribution setting. This includes, for example, the exponential sample lower bounds for tasks such as entropy estimation and independence testing [51]. In this work, we give efficient algorithms as well as lower bounds for two basic problems:

- Uniformity testing of unate distributions; and
- Testing unateness of an unknown distribution in the *subcube conditional model* introduced by Canonne, Ron, and Servedio [20] and Chakraborty, Fischer, Goldhirsh, and Matsliah [24].

In the subcube conditional model for distributions over $\{\pm 1\}^n$, an algorithm can query $\mathcal{D}$ with a subcube $\rho \in \{\pm 1, *\}^n$ and receive an independent sample $\boldsymbol{x} \sim \mathcal{D}$ conditioned on $\boldsymbol{x}_i = \rho_i$ for all i where $\rho_i \neq *$. This model provides a distributional analogue of the "membership query" from learning theory and property testing of Boolean functions, and has received considerable attention in recent years.

We note that for both of the problems we consider, an easy argument using the Chernoff bound together with a union bound over all $\sigma \in \{\pm 1\}^n$ (cf. Definition 1) allow us to reduce to the corresponding problems for monotone distributions with a multiplicative $O(n)$ sample/subcube query overhead. Our main algorithmic results improve on this naive approach for each task; we defer a technical overview of our results to Section 1.2 and survey related work in Section 1.3.

1.1.1 Uniformity Testing of Unate Distributions

Throughout, we write $\mathcal{U} = \mathcal{U}_n$ for the uniform distribution over $\{\pm 1\}^n$. Our first result gives an efficient uniformity tester for high-dimensional distributions under the promise of unateness:

▶ **Theorem 2.** *Let $\varepsilon \in (0,1)$. There is an algorithm,* `Unate-Uniformity` *(Algorithm 1) which, given i.i.d. sample access to an unknown unate distribution $\mathcal{D}$ over $\{\pm 1\}^n$, draws $\widetilde{O}(n^{3/2}/\varepsilon^2)$ samples and has the following performance guarantee:*

- *If $\mathcal{D} = \mathcal{U}$, then it outputs "accept" with probability $9/10$.*
- *If $\mathrm{d_{TV}}(\mathcal{D},\mathcal{U}) \geq \varepsilon$, then it outputs "reject" with probability $9/10$.*

Furthermore, it runs in $\widetilde{O}(n^{5/2}/\varepsilon^2)$ time.

Here $\mathrm{d_{TV}}(\cdot)$ refers to the total variation or statistical distance (Section 2.2). This result should be contrasted with the $\widetilde{O}(n/\varepsilon^2)$ bound for uniformity testing of monotone distributions due to Rubinfeld and Servedio [51].[2] At a high level, moving from monotone to unate distributions introduces an unknown orientation vector σ (cf. Definition 1); any naive strategy that first learns σ and then runs the [51] tester provably incurs an $\Omega(n^2)$ sample overhead. Our algorithm bypasses this bottleneck by never learning σ explicitly. Instead, we introduce a weak-orientation learning framework whose errors are provably weakly correlated across coordinates. This allows a small average bias to be amplified into a global signal using only $\widetilde{O}(n^{3/2})$ samples. We believe this structural result on limited correlations among marginals of unate distributions is of independent interest. We complement Theorem 2 with a matching lower bound:

[2] See also Adamaszek, Czumaj, and Sohler [2] who improved the sample complexity to $O(n/\varepsilon^2)$.

▶ **Theorem 3.** *Let $\mathcal{A}$ be any algorithm which, given i.i.d. sample access to an unknown distribution $\mathcal{D}$, has the performance guarantee from Theorem 2 with $\varepsilon = 1 - n^{-10}$. Then $\mathcal{A}$ must draw $\Omega(n^{3/2}/\log^2 n)$ samples from $\mathcal{D}$.*

Our proof of Theorem 3 extends the "monotone decomposition method" of Rubinfeld and Servedio [51] to the setting of unate distributions; see Section 1.2 for more details.

▶ Remark 4. One might hope that the additional power of the subcube conditional model could yield improved bounds for uniformity testing of unate distributions. Recall that uniformity testing of *arbitrary* distributions can be done using $\widetilde{O}(\sqrt{n})$ subcube queries [16]. Unfortunately, it turns out that the additional structure of unateness cannot beat this baseline: Chakrabarty, Chen, Ristic, Seshadhri, and Waingarten [21] have recently shown that even the easier problem of testing uniformity of monotone distributions in the subcube model already requires $\widetilde{\Omega}(\sqrt{n})$ queries.

1.1.2 Testing Unateness of Distributions

We now turn to the problem of testing whether an unknown distribution over $\{\pm 1\}^n$ is itself unate. In the standard access model where one receives i.i.d. samples from the unknown distribution being tester, this task is intractable: even for the simpler case of monotonicity, classical "birthday paradox" arguments imply that exponential-in-n sample complexity is unavoidable [4, 52]. It is therefore necessary to consider stronger access models to obtain meaningful algorithms.

Motivated by this, Chakrabarty, Chen, Ristic, Seshadhri, and Waingarten [21] studied monotonicity testing in the subcube conditional model, where the tester can query samples conditioned on arbitrary coordinate restrictions. Building on their results, we give an efficient algorithm for testing unateness in the same model.

▶ **Theorem 5.** *Let $\varepsilon \in (0, 1)$. There is an algorithm,* `Subcube-Unate` *(Algorithm 2) which, given subcube query access to an unknown distribution $\mathcal{D}$ over $\{\pm 1\}^n$, makes $\widetilde{O}(n^{3/2}/\varepsilon^2)$ queries and has the following performance guarantee:*

- *If $\mathcal{D}$ is unate, it outputs "accept" with probability at least $2/3$.*
- *If $\mathcal{D}$ is ε-far in TV distance from every unate distribution on $\{\pm 1\}^n$, then it outputs "reject" with probability at least $2/3$.*

We note that the algorithm of Theorem 5 also works in the weaker *coordinate oracle* model of Blanca et al. [13] where queries are only made on one-dimensional subcubes. We complement Theorem 5 with the following lower bound:

▶ **Theorem 6.** *There exists a constant $\varepsilon_0 > 0$ such that the following holds: If an algorithm tests whether an unknown distribution $\mathcal{D}$ over $\{\pm 1\}^n$ is unate or ε_0-far from unate by drawing Q_1 samples from $\mathcal{D}$ and making Q_2 (possibly adaptive) subcube queries on subcubes of dimensions $\leq d$, we have*

$$Q_1 + 2^d Q_2 \geq \Omega(n^{2/3}/\log^3 n).$$

We note that all prior lower bounds in the subcube model are based on product distributions as hard instances. Since every product distribution is trivially unate, these techniques cannot be applied to this setting. To the best of our knowledge, ours is the first lower bound that breaks this barrier; we return to this point in Section 1.4.

1.2 Main Ideas

We give a brief technical overview of our results. Although both of our upper bounds end up with the same asymptotic sample complexity of $\widetilde{O}(n^{3/2}/\varepsilon^2)$, that the ideas underlying each of them are quite different.

1.2.1 Testing Uniformity of Unate Distributions

Upper Bound. A natural first attempt is to adapt the Rubinfeld–Servedio [51] uniformity tester for monotone distributions to the unate setting. The idea would be to first learn the hidden "orientation vector" $\sigma^* \in \{\pm 1\}^n$ (cf. Definition 1) such that the reoriented distribution $\mathcal{D}^{\sigma^*}$, where $\boldsymbol{x} \sim \mathcal{D}^{\sigma^*}$ is obtained by drawing $\boldsymbol{y} \sim \mathcal{D}$ and setting $\boldsymbol{x}_i = \sigma_i^* \cdot \boldsymbol{y}_i$, is monotone. Since total variation distance is preserved under bijections, $\mathrm{d_{TV}}(\mathcal{D},\mathcal{U}) = \mathrm{d_{TV}}(\mathcal{D}^{\sigma^*},\mathcal{U})$. Thus, if σ^* were known, one could directly apply the $O(n/\varepsilon^2)$-sample uniformity tester of [51, 2].

However, exactly recovering σ^* can be prohibitively expensive. Consider, for example, a product distribution where each coordinate has mean $\pm\Theta(1/n)$. Such a distribution is $\Omega(1)$-far from uniform, but distinguishing $\mathbf{E}[x_i] = 0$ from $\mathbf{E}[x_i] = \pm\Theta(1/n)$ for each coordinate requires $\Omega(n^2)$ samples by Chernoff bounds. This naive approach of fully learning the orientation thus fails to give the desired sample complexity of $\widetilde{O}(n^{3/2})$.

Our algorithm circumvents this barrier by avoiding the need to learn σ^* exactly. Instead, we estimate a weakly correlated proxy for σ^* using only $\widetilde{O}(n^{3/2}/\varepsilon^2)$ samples. The key technical step (Lemma 11; which may be of independent interest) shows that for unate distributions, the coordinate-wise sign estimates cannot exhibit strong correlations: indeed, their covariances decay at rate $O(1/\sqrt{m})$ after m samples. This weak dependence is sufficient to amplify a small average bias across coordinates into a global signal, allowing us to test uniformity with $\widetilde{O}(n^{3/2})$ samples.

Lower Bound. The matching lower bound for uniformity testing is proved in the full version [45].

1.2.2 Testing Unate Distributions via Subcube Queries

Upper Bound. For the task of testing whether an unknown distribution is itself unate, we work in the *subcube conditional model*, following the recent work of Chakrabarty et al. [21]. At a high level, our algorithm adapts the classical *edge tester* for Boolean unateness [38] to the distributional setting. The central tool is a structural lemma of [21], based on a real-valued "directed" generalization of an isoperimetric inequality due to Talagrand [53], which controls the "edge bias" of monotone distributions. We refer the reader to Lemma 14 for a precise statement. Using this lemma, we show that if $\mathcal{D}$ is ε-far from unate, then there exists some coordinate i that simultaneously witnesses both "monotone" and "anti-monotone" violations. This can then be detected by making $\widetilde{O}(n^{3/2})$ subcube edge queries; the analysis requires an elementary but careful calculation.

Lower Bound. The lower bound for unateness testing in the subcube model is proved in the full version [45].

1.3 Related Work

Monotonicity of Boolean functions is among the most extensively studied properties in sublinear algorithms; we will not attempt to survey the very large body of results here and instead refer the reader to the comprehensive discussion from [21]. The problem of unateness

testing of Boolean functions was introduced alongside that of monotonicity testing in [38], and a sequence of works [38, 42, 23, 31, 32, 30] have pinned down its exact query complexity: $\widetilde{\Theta}(n^{2/3})$ queries are both necessary and sufficient to test unateness of Boolean functions.

A number of natural distribution testing tasks have exponential sample complexity in high-dimensional settings. This includes uniformity testing over $\{\pm 1\}^n$; see the surveys [15, 18] for more discussion on this. In order to overcome these lower bounds, many works either make structural assumptions on the distribution being tested, or assume stronger access models to the distribution. Examples of the former include monotonicity [51, 2], Bayesian networks [17, 35], Markov random fields [34, 8], various classes of structured truncations [40, 36, 37]; see also Section 7 of [15].

The subcube conditioning model, introduced by [20, 24, 10], takes the other route of assuming stronger access to the distribution being tested. This model has received much attention in recent years [16, 27, 47, 28, 3, 13, 21], and has also found applications beyond distribution testing to problems in learning theory [27, 12]. There has also been a nascent line of work on using the subcube conditioning model to make high-dimensional distribution testing practical [46, 49].

The works most relevant to our results are [51] and [21]. Rubinfeld and Servedio [51] establish an essentially tight bound of $\widetilde{\Theta}(n)$ samples for the problem of testing uniformity of an unknown monotone distribution, whereas Charkabarty et al. [21] give $\widetilde{\Theta}(n)$-query upper and lower bounds for testing monotonicity of an unknown distribution in the subcube model. Both our results – for uniformity testing as well as unateness testing – rely on ingredients going into the analyses of [51, 21] respectively.

1.4 Discussion

Our results raise a number of intriguing questions for further study; we highlight two compelling directions below.

Lower Bounds for Unateness Testing with Subcube Queries. Recall that Theorem 6 gives a $\widetilde{\Omega}(n^{2/3})$ query lower bound against algorithms that make $O(1)$-dimensional subcube queries. At present, all known lower-bounds in the subcube conditional model [16, 27, 21] construct hard instances that are product distributions, which are amenable to moment-matching techniques [27, 21]. However, since product distributions are automatically unate, none of these techniques can be lifted the unateness setting. Improving on Theorem 6 is thus likely to lead to new insights and techniques for proving lower bounds in the subcube conditional model.

***k*-Monotone Distributions.** Finally, one can consider distributional analogues of other structural generalizations of monotonicity. A natural candidate is the class of *k-monotone distributions*, where the distribution's mass function changes direction at most k times along any monotone path from -1^n to 1^n. The class of k-monotone functions has already been studied in the Boolean function setting [39, 19, 33], and k-monotone distributions could provide another structured setting for distribution testing beyond monotonicity.

2 Preliminaries

We use boldfaced letters such as (e.g. $\boldsymbol{x} \sim \{\pm 1\}^n$) to denote random variables. Unless explicitly stated otherwise, all probabilities and expectations will be with respect to the uniform distribution. Throughout, we will write $\mathcal{U}_n$ for the uniform distribution over $\{\pm 1\}^n$ and when the dimension n is clear from context we will simply write $\mathcal{U}$ instead.

For a distribution $\mathcal{D}$, it will be convenient to write $\mathcal{D}^{\otimes m}$ for the distribution of m independent draws from $\mathcal{D}$. We will use the following notation throughout:

▶ **Notation 7.** *Given a distribution $\mathcal{D}$ over $\{\pm 1\}^n$ and $\sigma \in \{\pm 1\}^n$, we write $\mathcal{D}^\sigma$ for the distribution over $\{\pm 1\}^n$ where a draw $\boldsymbol{x} \sim \mathcal{D}^\sigma$ is obtained by first drawing $\boldsymbol{y} \sim \mathcal{D}$ and then setting $\boldsymbol{x}_i \coloneqq \sigma_i \cdot \boldsymbol{y}_i$ for $i \in [n]$.*

2.1 The Berry–Esseen Theorem

We will write $N(0, I_d)$ for the d-dimensional standard Gaussian distribution, where I_d denotes the $d \times d$ identity matrix. We recall the (multi-dimensional) Berry–Esseen theorem (which will be used to prove Lemma 11):

▶ **Theorem 8** (Theorem 1.1 of [7]). *Let $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_m$ be i.i.d. random vectors in $\mathbb{R}^d$ distributed as $\boldsymbol{X}$ where $\mathbf{E}[\boldsymbol{X}] = 0$ and $\mathbf{E}[\boldsymbol{X}\boldsymbol{X}^\top] = I_d$. Then for any convex set $A \subseteq \mathbb{R}^d$, we have*

$$\left| \mathbf{Pr}\left[\frac{\sum_{i=1}^m \boldsymbol{X}_i}{\sqrt{m}} \in A \right] - \mathbf{Pr}[\boldsymbol{G} \in A] \right| \leq O(1) \cdot \frac{d^{1/4}}{\sqrt{m}} \cdot \mathbf{E}\left[\|\boldsymbol{X}\|^3 \right].$$

where $\boldsymbol{G} \sim N(0, I_d)$.

2.2 Distance Metrics Between Probability Distributions

We will identify a distribution $\mathcal{D}$ over a discrete domain Ω with its density function $\mathcal{D} : \Omega \to [0, 1]$. Recall that for two distributions $\mathcal{D}_1$, $\mathcal{D}_2$ over a discrete domain Ω, the *total variation* (or *statistical*) distance between $\mathcal{D}_1$ and $\mathcal{D}_2$ is

$$\mathrm{d}_{\mathrm{TV}}(\mathcal{D}_1, \mathcal{D}_2) = \frac{1}{2} \sum_{x \in \Omega} \left| \mathcal{D}_1(x) - \mathcal{D}_2(x) \right|.$$

We will frequently make use of the fact that total variation distance is invariant under bijections.

We also recall the *Kullback–Leibler (KL)* and χ^2-divergences:

$$\mathrm{d}_{\mathrm{KL}}(\mathcal{D}_1, \mathcal{D}_2) = \sum_{x \in \Omega} \mathcal{D}_1(x) \log \frac{\mathcal{D}_1(x)}{\mathcal{D}_2(x)} \qquad \text{and} \qquad \mathrm{d}_{\chi^2}(\mathcal{D}_1, \mathcal{D}_2) = \sum_{x \in \Omega} \frac{\left(\mathcal{D}_1(x) - \mathcal{D}_2(x)\right)^2}{\mathcal{D}_2(x)}$$

respectively. The following relationships are standard:

$$\mathrm{d}_{\mathrm{TV}}(\mathcal{D}_1, \mathcal{D}_2) \leq \sqrt{\frac{1}{2} \mathrm{d}_{\mathrm{KL}}(\mathcal{D}_1, \mathcal{D}_2)} \leq \sqrt{\frac{1}{2} \log\left(1 + \mathrm{d}_{\chi^2}(\mathcal{D}_1, \mathcal{D}_2)\right)}. \tag{1}$$

We will use the following version of the data processing inequality:

▶ **Fact 9.** *Let $\boldsymbol{X}_1$, $\boldsymbol{X}_2$ be random variables over the same domain. For any (possibly randomized) algorithm $\mathcal{A}$, we have $\mathrm{d}_{\mathrm{TV}}\big(\mathcal{A}(\boldsymbol{X}_1), \mathcal{A}(\boldsymbol{X}_2)\big) \leq \mathrm{d}_{\mathrm{TV}}(\boldsymbol{X}_1, \boldsymbol{X}_2)$.*

3 Testing Uniformity of Unate Distributions

We will prove Theorems 2 and 3 in this section, starting with the former.

3.1 Upper Bound

We first recall the principal technical lemma of [51]'s uniformity tester for monotone distributions:

▶ **Lemma 10** (Corollary 6 of [51]). *If $\mathcal{D}$ is a monotone distribution over $\{\pm 1\}^n$ that is ε-far from uniform, then*

$$\mathop{\mathbf{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\sum_{i=1}^{n}\boldsymbol{x}_i\right] \geq \varepsilon\,.$$

We now turn to our uniformity tester for unate distributions. First, note that if $\mathcal{D}$ is a unate distribution, then there exists some $\sigma^* \in \{\pm 1\}^n$ such that $\mathcal{D}^{\sigma^*}$ is a monotone distribution. In particular, note that we may take $\sigma_i^* = \mathrm{sign}(\mathbf{E}_{\boldsymbol{x}\sim\mathcal{D}}[\boldsymbol{x}_i])$. This suggests a natural algorithm: estimate σ^* by drawing m samples $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(m)}$ from $\mathcal{D}$ and setting

$$\boldsymbol{\sigma}_i := \mathrm{sign}\left(\sum_{j=1}^{m}\boldsymbol{x}_i^{(j)}\right),$$

and then run the [51] uniformity tester on the monotone distribution $\mathcal{D}^{\boldsymbol{\sigma}}$. Indeed, since

$$\mathrm{d}_{\mathrm{TV}}(\mathcal{D}^{\boldsymbol{\sigma}},\mathcal{U}) = \mathrm{d}_{\mathrm{TV}}(\mathcal{D}^{\boldsymbol{\sigma}},\mathcal{U}^{\boldsymbol{\sigma}}) = \mathrm{d}_{\mathrm{TV}}(\mathcal{D},\mathcal{U}),$$

replacing $\mathcal{D}$ with $\mathcal{D}^{\boldsymbol{\sigma}}$ preserves the uniformity (or distance from uniformity) of the distribution.

However, learning the whole sign vector σ^* can be too expensive. In particular, while Lemma 10 ensures that $\sum_{i=1}^{n}\mathbf{E}[\boldsymbol{x}_i]$ is large, it gives no guarantee on the magnitude of each individual $\mathbf{E}[\boldsymbol{x}_i]$. In particular, note that distinguishing $\mathbf{E}[\boldsymbol{x}_i] = 0$ (which corresponds to the uniform distribution) from $\mathbf{E}[\boldsymbol{x}_i] = \pm\Theta(1/n)$ requires $\Theta(n^2)$ samples per coordinate by standard Chernoff lower bounds. We bypass this obstacle by not insisting on the exact orientation and instead *weakly* learning the sign vector σ^*.

3.1.1 Warm-Up: A Sub-Quadratic Tester in a Special Case

As a warm-up, we show how to obtain a sub-quadratic tester in the hard instance just described above, where naively learning σ^* requires $\Omega(n^2)$ queries. It will be convenient to write

$$\mu_i := \mathop{\mathbf{E}}_{\boldsymbol{x}\sim\mathcal{D}}[\boldsymbol{x}_i]\,.$$

We make one simplifying assumption for now: $\mu_i = \frac{1}{n}$ holds for every $i \in [n]$, i.e., the unate distribution $\mathcal{D}$ is in fact monotone, and each coordinate has the same bias of $\Theta(1/n)$ towards $+1$.

Let $p > 0$ be a parameter that we will set later. By the assumption that $\mu_i = \frac{1}{n}$, it can be verified that $m = \Theta(n^2p^2)$ samples suffice for the "orientation step" to produce random signs $\boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_n \in \{\pm 1\}$ with the promise that

$$\mathbf{E}[\boldsymbol{\sigma}_i] \geq p \text{ for all } i \in [n]\,.$$

Equivalently, we correctly deduce that $\sigma_i^* = +1$ with probability $1/2 + \Omega(p)$, which is slightly better than a random guess.

Writing $\boldsymbol{X} \coloneqq \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\sigma}_i \in [-1,1]$, note that $\mathbf{E}[\boldsymbol{X}] \geq p$. Applying Markov's inequality to the non-negative random variable $1 - \boldsymbol{X}$ gives

$$\mathbf{Pr}\left[\boldsymbol{X} \leq \frac{p}{2}\right] = \mathbf{Pr}\left[1 - \boldsymbol{X} \geq 1 - \frac{p}{2}\right] \leq \frac{\mathbf{E}\left[1-\boldsymbol{X}\right]}{1-\frac{p}{2}} \leq \frac{1-p}{1-\frac{p}{2}} \leq 1 - \Omega(p)\,.$$

In other words, the event $\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\sigma}_i \geq \frac{p}{2}$ holds with probability $\Omega(p)$. Repeating this "orientation step" independently $O(1/p)$ times ensures that with constant probability, at least one draw satisfies $\boldsymbol{X} \geq p/2$. Then, we note that, conditioning on the realization of $\boldsymbol{\sigma} \in \{\pm 1\}^n$, the $\boldsymbol{\sigma}$-weighted Hamming weight $\sum_{i=1}^{n} \boldsymbol{\sigma}_i \boldsymbol{x}_i$ for $\boldsymbol{x} \sim \mathcal{D}$ has an expectation of exactly

$$\sum_{i=1}^{n} \boldsymbol{\sigma}_i \cdot \mathop{\mathbf{E}}_{\boldsymbol{x}\sim\mathcal{D}}\left[\boldsymbol{x}_i\right] = \sum_{i=1}^{n} \boldsymbol{\sigma}_i \cdot \mu_i = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\sigma}_i.$$

Therefore, by estimating $\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\sigma}_i$ to an additive error of $O(p)$ using $\widetilde{O}(n/p^2)$ additional samples, we obtain a tester with sample complexity (modulo polylogarithmic factors)

$$\frac{1}{p} \cdot n^2 p^2 + \frac{n}{p^2}. \tag{2}$$

Setting $p = n^{-1/3}$ gives a sample complexity of $\widetilde{O}(n^{5/3})$, which already improves upon the easy $\widetilde{O}(n^2)$ bound (cf. Section 1.1).

3.1.2 Main Technical Lemma

Recall from the warm-up that the sign estimates $\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2, \ldots, \boldsymbol{\sigma}_n$ satisfy $\mathbf{E}\left[\boldsymbol{\sigma}_i\right] \geq p$, which implies

$$\mathop{\mathbf{Pr}}_{\boldsymbol{\sigma}}\left[\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\sigma}_i \geq \Omega(p)\right] \geq \Omega(p)\,.$$

Now, note that if we were able to replace the R.H.S. above with $\Omega(1)$ instead of $\Omega(p)$, then this stronger lower bound would imply that $O(1)$ independent sign estimates would suffice for "catching" the $\Omega(p)$ bias. Then, the sample complexity (up to polylogarithmic factors) would be $n^2p^2 + \frac{n}{p^2}$ (cf. Equation (2)), which reduces to $\widetilde{O}(n^{3/2})$ if we set $p = n^{-1/4}$.

We first note that the property $\mathbf{E}\left[\boldsymbol{\sigma}_i\right] \geq p$ alone does not imply the desired high-probability bound, namely,

$$\mathbf{Pr}\left[\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\sigma}_i \geq \Omega(p)\right] \geq \Omega(1).$$

Suppose for the sake of simplicity that n is even, and imagine that $\boldsymbol{\sigma} \in \{\pm 1\}^n$ is drawn from the following distribution:

- With probability p, $\boldsymbol{\sigma} = (+1, +1, \ldots, +1)$.
- With the remaining probability $1-p$, $\boldsymbol{\sigma}$ is chosen as a uniformly random permutation of $n/2$ copies of $+1$ and $n/2$ copies of -1.

It can be easily verified that the above distribution satisfies $\mathbf{E}\left[\boldsymbol{\sigma}_i\right] = p$, yet $\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\sigma}_i > 0$ only holds with probability p.

Therefore, the crux is to show that the $\boldsymbol{\sigma}_i$'s cannot be too correlated (as in the example above). Concretely, proving the following will suffice for our purposes:

Algorithm 1 The `Unate-Uniformity` algorithm.

Input: Access to i.i.d. samples from unate $\mathcal{D}$, distance parameter $\varepsilon \in (0,1)$

Output: "Accept" or "reject"

`Unate-Uniformity`$(\mathcal{D}, \varepsilon)$:

1. Repeat the following $O(1)$ times:
   a. Draw $m_1 := \Theta\left(\frac{n^{3/2}}{\varepsilon^2}\right)$ samples $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m_1)} \sim \mathcal{D}$.
   b. Compute $\boldsymbol{\sigma} \in \{\pm 1\}^n$ where for $i \in [n]$ we have

$$\boldsymbol{\sigma}_i := \operatorname{sign}\left(\sum_{j=1}^{m_1} \boldsymbol{x}_i^{(j)}\right) \in \{\pm 1\}\,.$$

   c. Draw $m_2 := \Theta\left(\frac{n^{3/2}}{\varepsilon^2} \log\left(\frac{n}{\varepsilon}\right)\right)$ samples $\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(m_2)} \sim \mathcal{D}$.
   d. For each $j \in [m_2]$, let $\boldsymbol{w}_j := \sum_{i=1}^n \boldsymbol{\sigma}_i \cdot \boldsymbol{y}_i^{(j)}$ denote the $\boldsymbol{\sigma}$-weighted Hamming weight of $\boldsymbol{y}^{(j)}$.
   e. If $|\boldsymbol{w}_j| \geq \Theta(\sqrt{n \log(n/\varepsilon)})$ holds for any $j \in [m_2]$, halt and output "reject."
   f. If $\frac{1}{m_2} \sum_{j=1}^{m_2} \boldsymbol{w}_j \geq \Theta(\varepsilon/n^{1/4})$, then halt and output "reject."
2. If the algorithm has not rejected yet, output "accept."

▶ **Lemma 11.** *Suppose $\mathcal{D}$ is a unate distribution on $\{\pm 1\}^n$. Suppose $\boldsymbol{x}^{(1)}, \ldots \boldsymbol{x}^{(m)} \sim \mathcal{D}$ and let*

$$\boldsymbol{\sigma}_i := \operatorname{sign}\left(\sum_{j=1}^{m} \boldsymbol{x}_i^{(j)}\right).$$

For any $1 \leq i < j \leq n$, it holds that

$$|\operatorname{\mathbf{Cov}}[\boldsymbol{\sigma}_i, \boldsymbol{\sigma}_j]| = O\left(\frac{1}{\sqrt{m}}\right).$$

It is readily verified that Lemma 11 fails for non-unate distributions: consider, for example, the two-point distribution $\mathcal{D}(+1^n) = \mathcal{D}(-1^n) = 0.5$. We defer the proof of Lemma 11 to Section 3.1.4, and first show why it implies the correctness of Algorithm 1.

3.1.3 Proof of Theorem 2

We will require the following lemma due to Qiao and Valiant [50]:

▶ **Lemma 12** (Lemma B.2 of [50])**.** *Suppose $k \in \mathbb{N}$ and $\delta \in (0, 1/15\sqrt{k})$. Let P and Q be two distributions on the same support with $\mathrm{d}_{\mathrm{TV}}(P, Q) \geq \delta$. Then*

$$\mathrm{d}_{\mathrm{TV}}(P^{\otimes k}, Q^{\otimes k}) \geq \frac{\delta\sqrt{k}}{15}\,.$$

Proof of Theorem 2. Note that the sample complexity and runtime are evident from Algorithm 1. Next, note that if $\mathcal{D} = \mathcal{U}$, then $\mathcal{D}^\sigma = \mathcal{U}$ for every $\sigma \in \{\pm 1\}^n$. It now follows from Theorem 4 of [51] that `Unate-Uniformity` will output "accept" with probability at least 9/10. In particular, note that in this case, Steps 1(c) through 1(f) of `Unate-Uniformity` are identical to the `TestUniform` algorithm of [51] run on $\mathcal{U}$.

We now show how Lemma 11 implies the soundness of `Unate-Uniformity`. Suppose that the unate distribution $\mathcal{D}$ is ε-far from uniform. As before, we write

$$\mu_i \coloneqq \mathop{\mathbf{E}}_{\boldsymbol{x}\sim\mathcal{D}}[\boldsymbol{x}_i]\,.$$

Since $\mathcal{D}$ is unate, it follows that there exists some $\sigma^* \in \{\pm 1\}^n$ such that $\mathcal{D}^{\sigma^*}$ is monotone. In particular, we can take $\sigma_i^* = \operatorname{sign}(\mu_i)$, and so we have $\sigma_i^*\mu_i = |\mu_i| \geq 0$. Since the TV distance is preserved under bijections, $\mathcal{D}^{\sigma^*}$ is also ε-far from uniform. Applying Lemma 10 to $\mathcal{D}^{\sigma^*}$ gives

$$\|\mu\|_1 = \sum_{i=1}^n \sigma_i^*\mu_i = \mathop{\mathbf{E}}_{\boldsymbol{x}\sim\mathcal{D}^{\sigma^*}}\left[\sum_{i=1}^n \boldsymbol{x}_i\right] \geq \varepsilon\,.$$

Recall that we use a sample of size $m_1 = \Theta(n^{3/2}/\varepsilon^2)$ in Step 1(a) and 1(b) to compute $\boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_n \in \{\pm 1\}$. Set a threshold $\theta = \frac{\varepsilon}{100n}$. For every $i \in [n]$ that satisfies $|\mu_i| \geq \theta$, we will now show that

$$\mathbf{E}\left[\sigma_i^*\boldsymbol{\sigma}_i\right] \geq \Omega\left(\frac{\varepsilon}{n}\cdot\frac{n^{3/4}}{\varepsilon}\right) = \Omega(n^{-1/4})\,, \tag{3}$$

using Lemma 12. In more detail, let P be the distribution of $\sigma_i^*\boldsymbol{x}_i$ for $\boldsymbol{x}\sim\mathcal{D}$ and Q be $-P$, i.e. $\boldsymbol{y}\sim Q$ is obtained by drawing $\boldsymbol{y}'\sim P$ and setting $\boldsymbol{y} = -\boldsymbol{y}'$. Note that

$$\Pr_{\boldsymbol{y}\sim\mathcal{P}}[\boldsymbol{y}=+1] = \frac{1+|\mu_i|}{2} \qquad \text{and} \qquad \Pr_{\boldsymbol{y}\sim\mathcal{P}}[\boldsymbol{y}=-1] = \frac{1-|\mu_i|}{2}\,,$$

with swapped probabilities for Q. It follows that $\mathrm{d_{TV}}(P,Q) = |\mu_i| \geq \frac{\varepsilon}{100n}$; note also that

$$\frac{\varepsilon}{100n} \leq \frac{1}{15\sqrt{m_1}}$$

for appropriate choice of hidden constant in m_1. In particular, applying Lemma 12 gives

$$\mathrm{d_{TV}}(P^{\otimes m_1}, Q^{\otimes m_1}) \geq \Omega\left(\frac{\varepsilon}{n}\cdot\frac{n^{3/4}}{\varepsilon}\right) = \Omega(n^{-1/4})\,.$$

In order to establish Equation (3), it suffices to show that $\mathrm{d_{TV}}(P^{\otimes m_1}, Q^{\otimes m_1}) = \Theta(1)\cdot\mathbf{E}[\sigma_i^*\boldsymbol{\sigma}_i]$. Since P and Q are Bernoulli random variables, their likelihood ratio is monotone in the number of successes (i.e., $+1$ draws), and so by the Neyman–Pearson lemma, the TV-distance maximizing set is $\{\sum_\ell \boldsymbol{y}^{(\ell)} \geq 0\}$. We may assume that m_1 is odd (to avoid handling ties), and so we get

$$\begin{aligned}
\mathrm{d_{TV}}(P^{\otimes m_1}, Q^{\otimes m_1}) &= P^{\otimes m_1}\left(\sum_{\ell=1}^{m_1}\boldsymbol{y}^{(\ell)} \geq 0\right) - Q^{\otimes m_1}\left(\sum_{\ell=1}^{m_1}\boldsymbol{y}^{(\ell)} \geq 0\right) \\
&= P^{\otimes m_1}\left(\sum_{\ell=1}^{m_1}\boldsymbol{y}^{(\ell)} \geq 0\right) - P^{\otimes m_1}\left(\sum_{\ell=1}^{m_1}\boldsymbol{y}^{(\ell)} < 0\right) \qquad (4)\\
&= \mathbf{E}[\sigma_i^*\boldsymbol{\sigma}_i]\,,
\end{aligned}$$

from which Equation (3) follows readily. Note that Equation (4) relies on the fact that $P = -Q$. Finally, note also that the same argument shows that $\mathbf{E}\left[\sigma_i^*\boldsymbol{\sigma}_i\right] \geq 0$ for every $i \in [n]$, even if $|\mu_i| < \theta$.

Let $\boldsymbol{X} \coloneqq \sum_{i=1}^{n} \mu_i \cdot \boldsymbol{\sigma}_i = \sum_{i=1}^{n} |\mu_i| \cdot (\sigma_i^* \boldsymbol{\sigma}_i)$. We have

$$\mathbf{E}[\boldsymbol{X}] \geq \sum_{i \in [n]:|\mu_i| \geq \theta} |\mu_i| \cdot \mathbf{E}[\sigma_i^* \boldsymbol{\sigma}_i] \geq \Omega(n^{-1/4}) \cdot \sum_{i \in [n]:|\mu_i| \geq \theta} |\mu_i| = \Omega(n^{-1/4} \cdot \|\mu\|_1).$$

The first step above holds since both $|\mu_i|$ and $\mathbf{E}\left[\sigma_i^* \boldsymbol{\sigma}_i\right]$ are non-negative for every $i \in [n]$. The second step applies $|\mu_i| \geq \theta \implies \mathbf{E}\left[\sigma_i^* \boldsymbol{\sigma}_i\right] \geq \Omega(n^{-1/4})$. The third step holds since our choice of $\theta = \frac{\varepsilon}{100n}$ and the fact $\|\mu\|_1 \geq \varepsilon$ together imply

$$\sum_{i \in [n]:|\mu_i| \geq \theta} |\mu_i| \geq \sum_{i=1}^{n} |\mu_i| - n \cdot \theta = \|\mu\|_1 - \frac{\varepsilon}{100} \geq \Omega(\|\mu\|_1).$$

It remains to show that

$$\mathbf{Var}[\boldsymbol{X}] \leq O(n^{-1/2} \cdot \|\mu\|_1^2).$$

By Chebyshev's inequality, the variance bound above would imply that, with probability $\Omega(1)$,

$$\boldsymbol{X} \geq \mathbf{E}[\boldsymbol{X}] - O(\sqrt{\mathbf{Var}[\boldsymbol{X}]}) \geq \Omega(n^{-1/4} \cdot \|\mu\|_1) \geq \Omega(\varepsilon / n^{1/4}),$$

where the second step applies the following observations:

- $\mathbf{E}[\boldsymbol{X}] \geq \Omega(n^{-1/4} \cdot \|\mu\|_1)$, where the hidden constant in $\Omega(\cdot)$ is lower bounded by a universal constant when $m_1 = \Theta(n^{3/2}/\varepsilon^2)$ is sufficiently large.
- We will show that $\mathbf{Var}[\boldsymbol{X}] \leq O(n^{-1/2} \cdot \|\mu\|_1^2)$, where the hidden constant in $O(\cdot)$ goes to zero as the hidden constant in $m_1 = \Theta(n^{3/2}/\varepsilon^2)$ increases.
- Therefore, for some careful choice of m_1 in the algorithm, the difference $\mathbf{E}[\boldsymbol{X}] - O(\sqrt{\mathbf{Var}[\boldsymbol{X}]})$ is positive and on the order of $\Omega(n^{-1/4} \cdot \|\mu\|_1)$.

By a Chernoff bound, we can then catch this $\Omega(\varepsilon/n^{1/4})$ bias using $\widetilde{O}\left(\frac{n}{(\varepsilon/n^{1/4})^2}\right) = \widetilde{O}(n^{3/2}/\varepsilon^2)$ additional samples in Step 1(f) of the algorithm.

By Lemma 11, we have

$$\mathbf{Var}[\boldsymbol{X}] = \sum_{i=1}^{n} \mu_i^2 \mathbf{Var}[\boldsymbol{\sigma}_i] + 2 \sum_{1 \leq i < j \leq n} \mu_i \mu_j \mathbf{Cov}[\boldsymbol{\sigma}_i, \boldsymbol{\sigma}_j] \leq \sum_{i=1}^{n} \mu_i^2 \mathbf{Var}[\boldsymbol{\sigma}_i] + O(n^{-1/2}) \cdot \|\mu\|_1^2.$$

The second term above is exactly the desired upper bound. For the first term, we note that

$$\mathbf{Var}[\boldsymbol{\sigma}_i] = e^{-\Omega(m_1 \mu_i^2)},$$

so each term $\mu_i^2 \mathbf{Var}[\sigma_i]$ is at most $O(1/m_1)$. Recalling $m_1 = \Theta(n^{3/2}/\varepsilon^2)$ and $\|\mu\|_1 \geq \varepsilon$, we conclude that

$$\sum_{i=1}^{n} \mu_i^2 \mathbf{Var}[\boldsymbol{\sigma}_i] \leq O(n/m_1) = O(\varepsilon^2/\sqrt{n}) \leq O(n^{-1/2} \cdot \|\mu\|_1^2),$$

which proves the desired variance bound $\mathbf{Var}[\boldsymbol{X}] \leq O(n^{-1/2} \cdot \|\mu\|_1^2)$ and completes the proof. ◀

3.1.4 Proof of Lemma 11

Without loss of generality, we prove the lemma when the unate distribution $\mathcal{D}$ is monotone; the more general statement then follows from the observation that flipping the ith coordinate of $\mathcal{D}$ leads to a flip in the distribution of $\boldsymbol{\sigma}_i$, which preserves the magnitude of the covariances.

For $i \in [n]$, we define

$$\boldsymbol{S}_i := \sum_{\ell=1}^{m} \boldsymbol{x}_i^{(\ell)} \qquad \text{and} \qquad \mu_i := \mathop{\mathbf{E}}_{\boldsymbol{x}\sim\mathcal{D}}[\boldsymbol{x}_i]\,.$$

In particular, we have $\boldsymbol{\sigma}_i = \operatorname{sign}(\boldsymbol{S}_i) = 2\cdot\mathbf{1}\{\boldsymbol{S}_i \geq 0\} - 1$, and so

$$\mathbf{Cov}[\boldsymbol{\sigma}_i, \boldsymbol{\sigma}_j] = 4\big(\mathbf{Pr}[\boldsymbol{S}_i \geq 0, \boldsymbol{S}_j \geq 0] - \mathbf{Pr}[\boldsymbol{S}_i \geq 0]\,\mathbf{Pr}[\boldsymbol{S}_j \geq 0]\big)\,. \tag{5}$$

We will prove Lemma 11 with $i = 1$ and $j = 2$; that is, we will show that

$$\mathbf{Cov}[\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2] = O\left(\frac{1}{\sqrt{m}}\right).$$

Define

$$p_{00} := \mathop{\mathbf{Pr}}_{\boldsymbol{x}\sim\mathcal{D}}\left[(\boldsymbol{x}_1, \boldsymbol{x}_2) = (-1,-1)\right], \qquad p_{10} := \mathop{\mathbf{Pr}}_{\boldsymbol{x}\sim\mathcal{D}}\left[(\boldsymbol{x}_1, \boldsymbol{x}_2) = (+1,-1)\right],$$
$$p_{01} := \mathop{\mathbf{Pr}}_{\boldsymbol{x}\sim\mathcal{D}}\left[(\boldsymbol{x}_1, \boldsymbol{x}_2) = (-1,+1)\right], \qquad p_{11} := \mathop{\mathbf{Pr}}_{\boldsymbol{x}\sim\mathcal{D}}\left[(\boldsymbol{x}_1, \boldsymbol{x}_2) = (+1,+1)\right].$$

Thanks to monotonicity of $\mathcal{D}$, we have $p_{00} \leq p_{10} \leq p_{11}$ and $p_{00} \leq p_{01} \leq p_{11}$. It also follows that $\mu_1 = 1 - 2(p_{00} + p_{01}) \geq 0$ and $\mu_2 = 1 - 2(p_{00} + p_{10}) \geq 0$. The proof considers two cases depending on the value of $\mu_1 + \mu_2$:

Case 1: $\mu_1 + \mu_2 > 1/2$

We record the following easy bound on $\mathbf{Cov}[\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2]$:

$$\big|\mathbf{Cov}[\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2]\big| \leq 4\min\left\{\mathbf{Pr}\left[\boldsymbol{S}_1 < 0\right], \mathbf{Pr}\left[\boldsymbol{S}_2 < 0\right]\right\}. \tag{6}$$

Assuming Equation (6), a direct application of the Hoeffding bound gives

$$\mathbf{Pr}[\boldsymbol{S}_i < 0] \leq e^{-m\mu_i^2/2}\,, \qquad \text{and so} \qquad |\,\mathbf{Cov}(s_1, s_2)| \leq 4e^{-m/32} \ll O\left(\frac{1}{\sqrt{m}}\right),$$

where we make use of the fact that $\min\{\mu_1, \mu_2\} \geq 1/4$ by the assumption $\mu_1 + \mu_2 > 1/2$.

We now justify Equation (6). It will be convenient to write

$$A := \mathbf{Pr}[\boldsymbol{S}_1 \geq 0, \boldsymbol{S}_2 \geq 0]\,, \qquad B := \mathbf{Pr}[\boldsymbol{S}_1 \geq 0]\,, \qquad \text{and} \qquad C = \mathbf{Pr}[\boldsymbol{S}_2 \geq 0]\,.$$

Note that $\mathbf{E}[\boldsymbol{\sigma}_1] = 2B - 1$ and $\mathbf{E}[\boldsymbol{\sigma}_2] = 2C - 1$. Writing $D := 1 + A - B - C$, it is readily checked that $D = \mathbf{Pr}[\boldsymbol{S}_1 < 0, \boldsymbol{S}_2 < 0]$. It follows from Equation (5) that $\mathbf{Cov}[\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2] = 4(A - BC)$. In particular, we have

$$\frac{1}{4}\big|\mathbf{Cov}[\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2]\big| = |A - BC|\,.$$

Now,

- If $A \geq BC$, then $A - BC \leq C(1 - B) \leq 1 - B$.
- If $A < BC$, then $BC - A \leq C - (B + C - 1) = 1 - B$, since $D \geq 0$ gives $A \geq B + C - 1$.

It follows that $|\,\mathbf{Cov}[\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2]| \leq 4(1 - B) = 4\,\mathbf{Pr}[\boldsymbol{S}_1 < 0]$. By symmetry, the same holds with B and C swapped, which yields Equation (6).

Case 2: $\mu_1 + \mu_2 \le 1/2$

In this case, we will control $\mathbf{Cov}(\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$ via a Gaussian approximation to the sums $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$. First, let $\kappa := \mathbf{Cov}[\boldsymbol{x}_1, \boldsymbol{x}_2]$ for $\boldsymbol{x} \sim \mathcal{D}$. Note that $\kappa = p_{00} + p_{11} - p_{01} - p_{10} - \mu_1\mu_2$.

We will first control the covariance of the Gaussian approximation. Consider the standardized random vector

$$\boldsymbol{T} = (\boldsymbol{T}_1, \boldsymbol{T}_2) := \frac{1}{\sqrt{m}}(\boldsymbol{S}_1 - m\mu_1, \boldsymbol{S}_2 - m\mu_2) \qquad \text{and let} \qquad \Sigma := \begin{pmatrix} \lambda_1^2 & \kappa \\ \kappa & \lambda_2^2 \end{pmatrix}$$

where $\lambda_i^2 = 1 - \mu_i^2$. We will sometimes write $\mu = (\mu_1, \mu_2)$. Finally, let ρ be the correlation between $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$, that is,

$$\rho := \frac{\kappa}{\lambda_1\lambda_2}\,.$$

Note that by the central limit theorem, $\boldsymbol{T} \to N(0, \Sigma)$ (in distribution) as $m \to \infty$. Following Equation (5), the Gaussian approximation of the correlation is

$$\mathbf{Cov}_{\approx} := 4\Big(\mathbf{Pr}\left[\boldsymbol{Z}_1 \ge -\sqrt{m}\mu_1, \boldsymbol{Z}_2 \ge -\sqrt{m}\mu_1\right] - \mathbf{Pr}\left[\boldsymbol{Z}_1 \ge -\sqrt{m}\mu_1\right]\mathbf{Pr}\left[\boldsymbol{Z}_2 \ge -\sqrt{m}\mu_2\right]\Big).$$

Note that $\mathbf{Cov}_{\approx} = \mathbf{Cov}[\boldsymbol{\tau}_1, \boldsymbol{\tau}_2]$ where $\boldsymbol{\tau}_i = \mathrm{sign}(\boldsymbol{Z}_i - \sqrt{m}\mu_i)$. Next, we will show that $\mathbf{Cov}_{\approx} = O(m^{-1/2})$ via known estimates for computing bi-variate normal probabilities. In particular, applying Equation 3.6 of [48] gives

$$\begin{aligned}
|\,\mathbf{Cov}_{\approx}\,| &= 4\left|\frac{1}{2\pi}\int_0^{\rho}\frac{1}{\sqrt{1-z^2}}\exp\Big(-\frac{1}{2}\frac{(\sqrt{m}\frac{\mu_1}{\lambda_1})^2 + (\sqrt{m}\frac{\mu_2}{\lambda_2})^2 - 2(\sqrt{m}\frac{\mu_1}{\lambda_1})(\sqrt{m}\frac{\mu_2}{\lambda_2})z}{1-z^2}\Big)\,dz\right| \\
&\le \frac{2}{\pi}\int_0^{|\rho|}\frac{1}{\sqrt{1-z^2}}\exp\Big(-\frac{m}{2}\left(\frac{(\frac{\mu_1}{\lambda_1} - \frac{\mu_2}{\lambda_2})^2}{1-z^2} + \frac{2\frac{\mu_1}{\lambda_1}\frac{\mu_2}{\lambda_2}}{1+z}\right)\Big)\,dz \qquad (7) \\
&\le \frac{2}{\pi}\int_0^{|\rho|}\frac{1}{\sqrt{1-z^2}}\exp\Big(-\frac{m}{8}(\frac{\mu_1}{\lambda_1} + \frac{\mu_2}{\lambda_2})^2\Big)\,dz \qquad (8) \\
&\le \frac{2}{\pi}\sin^{-1}|\rho|\exp\Big(-\frac{m}{8}(\mu_1+\mu_2)^2\Big) \\
&\le |\rho|\exp\Big(-\frac{m}{8}(\mu_1+\mu_2)^2\Big)\,, \qquad (9)
\end{aligned}$$

where Equation (7) relies on the fact that the integrand is larger on the positive z side, Equation (8) uses

$$\frac{(\frac{\mu_1}{\lambda_1} - \frac{\mu_2}{\lambda_2})^2}{1-z^2} + \frac{2\frac{\mu_1}{\lambda_1}\frac{\mu_2}{\lambda_2}}{1+z} \ge (\frac{\mu_1}{\lambda_1} - \frac{\mu_2}{\lambda_2})^2 + \frac{\mu_1}{\lambda_1}\frac{\mu_2}{\lambda_2} = \frac{1}{4}(\frac{\mu_1}{\lambda_1} + \frac{\mu_2}{\lambda_2})^2 + \frac{3}{4}(\frac{\mu_1}{\lambda_1} - \frac{\mu_2}{\lambda_2})^2\,,$$

and Equation (9) relies on $\lambda_i \le 1$ and $\sin^{-1}(\theta) \le \frac{\pi\theta}{2}$ for $\theta \ge 0$.

We record the following lemma:

▶ **Lemma 13.** *For monotone distributions, if $\mu_1 + \mu_2 \le \frac{1}{2}$, then $|\rho| \le \mu_1 + \mu_2$.*

Proof. Parametrize the probabilities of the monotone distribution as

$$p_{00} = a, \quad p_{01} = a + b, \quad p_{10} = a + c, \quad p_{11} = a + d.$$

Then $\mu_1 = d + c - b$, $\mu_2 = d + b - c$, so $\mu_1 + \mu_2 = 2d$ implies $d \leq \frac{1}{4}$. Now

$$\begin{aligned}
|\rho| &= \frac{|(d-b-c) - (d+c-b)(d+b-c)|}{\sqrt{(1-\mu_1^2)(1-\mu_2^2)}} \\
&\leq \frac{2}{\sqrt{3}} \left|(d-b-c) + ((b-c)^2 - d^2)\right| && (10) \\
&\leq \frac{2}{\sqrt{3}}(d + d^2) && (11) \\
&= 2d \cdot \frac{1+d}{\sqrt{3}} \leq 2d = \mu_1 + \mu_2
\end{aligned}$$

where in Equation (10) we use that $(1-\mu_1^2)(1-\mu_2^2) \geq \frac{3}{4}$, and in Equation (11) we note that $|d - b - c| \leq d$ and $|b - c| \leq d$. ◀

Combining Equation (9) with Lemma 13, we get

$$|\operatorname{\mathbf{Cov}}_{\approx}| \leq (\mu_1 + \mu_2) \exp\left(-\frac{m}{8}(\mu_1 + \mu_2)^2\right) = O\left(\frac{1}{\sqrt{m}}\right). \tag{12}$$

Next, we will control the error in the Gaussian approximation to $\mathbf{Cov}[\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2]$ itself by applying the multivariate Berry-Eseeen central limit theorem (cf. Theorem 8). It is readily verified using Equation (5) and the triangle inequality that

$$\frac{1}{4}\left(\mathbf{Cov}[\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2] - \mathbf{Cov}_{\approx}\right) \leq \sum_{i=1}^{3} |\alpha_i - \beta_i|$$

where

$$\alpha_1 = \mathbf{Pr}[\boldsymbol{S}_1 \geq 0, \boldsymbol{S}_2 \geq 0], \quad \alpha_2 = \mathbf{Pr}[\boldsymbol{S}_1 \geq 0], \quad \alpha_3 = \mathbf{Pr}[\boldsymbol{S}_2 \geq 0],$$

$$\beta_1 = \mathbf{Pr}\left[\boldsymbol{Z}_1 \geq -\sqrt{m}\mu_1, \boldsymbol{Z}_2 \geq -\sqrt{m}\mu_2\right], \quad \beta_2 = \mathbf{Pr}\left[\boldsymbol{Z}_1 \geq -\sqrt{m}\mu_1\right], \quad \beta_3 = \mathbf{Pr}\left[\boldsymbol{Z}_2 \geq -\sqrt{m}\mu_2\right].$$

Theorem 8 combined with the fact that both $\mu_1, \mu_2 \leq 1/2$ immediately gives $|\alpha_2 - \beta_2|$, $|\alpha_3 - \beta_3| = O(m^{-1/2})$. In more detail, Theorem 8 gives

$$|\alpha_2 - \beta_2| = O\left(\frac{1}{\sqrt{m}} \cdot \frac{[|\boldsymbol{S}_1 - \mu_1|^3]}{\mathbf{E}[(\boldsymbol{S}_1 - \mu_1)^2]^{3/2}}\right) = O\left(\frac{1}{\sqrt{m}} \cdot \frac{(1+\mu_1)^3]}{(1-\mu_1^2)^{3/2}}\right) \leq O\left(\frac{1}{\sqrt{m}}\right).$$

An identical argument gives the same bound for $|\alpha_3 - \beta_3|$. Finally, in order to show that $|\alpha_1 - \beta_1| = O(m^{-1/2})$ using Theorem 8, it suffices to establish that $\mathbf{E}[\|\Sigma^{-1/2}(\boldsymbol{S} - \mu)\|_2^3] = O(1)$, where $\boldsymbol{S} = (\boldsymbol{S}_1, \boldsymbol{S}_2)$. To see this, first note that

$$\Sigma^{-1/2} = \frac{1}{2}\begin{bmatrix} \alpha & \beta \\ \beta & \alpha \end{bmatrix}\begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}^{-1}$$

where $\alpha = \frac{1}{\sqrt{1+\rho}} + \frac{1}{\sqrt{1-\rho}}$ and $\beta = \frac{1}{\sqrt{1+\rho}} - \frac{1}{\sqrt{1-\rho}}$. Now,

$$\begin{aligned}
\|\Sigma^{-1/2}(\boldsymbol{S}-\mu)\|_2^2 &= \frac{1}{4}\left((\alpha\lambda_1^{-1}(\boldsymbol{S}_1-\mu_1)+\beta\lambda_2^{-1}(\boldsymbol{S}_2-\mu_2))^2+(\beta\lambda_1^{-1}(\boldsymbol{S}_1-\mu_1)+\alpha\lambda_2^{-1}(\boldsymbol{S}_2-\mu_2))^2\right)\\
&= \frac{1}{1-\rho^2}\left(\lambda_1^{-2}(\boldsymbol{S}_1-\mu_1)^2+\lambda_2^{-2}(\boldsymbol{S}_2-\mu_2)^2-\rho\lambda_1^{-1}\lambda_2^{-1}(\boldsymbol{S}_1-\mu_1)(\boldsymbol{S}_2-\mu_2)\right) \qquad (13)\\
&\le \frac{4}{1-\rho^2}\left(\lambda_1^{-2}+\lambda_2^{-2}+\rho\lambda_1^{-1}\lambda_2^{-1}\right) \qquad (14)\\
&= 4\cdot\frac{2-(\mu_1^2+\mu_2^2)+\kappa}{(1-\mu_1^2)(1-\mu_2^2)-\kappa^2}\\
&\le 4\cdot\frac{2-\frac{1}{2}+\frac{1}{4}}{\frac{3}{4}-\left(\frac{4}{9}\right)^2} < 16, \qquad (15)
\end{aligned}$$

where in Equation (13) we note that $\alpha^2+\beta^2=\frac{4}{1-\rho^2}$ and $2\alpha\beta=-\frac{4\rho}{1-\rho^2}$, in Equation (14) we use $|X_i-\mu_i|\le 2$, and in Equation (15) we recall $(1-\mu_1^2)(1-\mu_2^2)\ge\frac{3}{4}$ and $-\frac{4}{9}\le\kappa\le\frac{1}{4}$. The result follows immediately. ◀

3.2 Lower Bound

Turning to lower bounds for uniformity testing of unate distribution, we will establish the following:

▶ **Theorem 3.** *Let $\mathcal{A}$ be any algorithm which, given i.i.d. sample access to an unknown distribution $\mathcal{D}$, has the performance guarantee from Theorem 2 with $\varepsilon = 1-n^{-10}$. Then $\mathcal{A}$ must draw $\Omega(n^{3/2}/\log^2 n)$ samples from $\mathcal{D}$.*

In particular, Theorem 3 implies that the algorithm from Section 3.1 has essentially optimal sample complexity. The proof is deferred to the full version [45].

4 Testing Unateness with Subcube Conditioning

We now turn to the proofs of Theorems 5 and 6, starting with the former.

4.1 Upper Bound

We start by recalling Theorem 5:

▶ **Theorem 5.** *Let $\varepsilon\in(0,1)$. There is an algorithm,* `Subcube-Unate` *(Algorithm 2) which, given subcube query access to an unknown distribution $\mathcal{D}$ over $\{\pm1\}^n$, makes $\widetilde{O}(n^{3/2}/\varepsilon^2)$ queries and has the following performance guarantee:*

- *If $\mathcal{D}$ is unate, it outputs "accept" with probability at least $2/3$.*
- *If $\mathcal{D}$ is ε-far in TV distance from every unate distribution on $\{\pm1\}^n$, then it outputs "reject" with probability at least $2/3$.*

4.1.1 Preliminaries from [21]

Chakrabarty, Chen, Ristic, Seshadhri, and Waingarten [21] gave an $\widetilde{O}(n/\varepsilon^2)$-query algorithm for monotonicity testing of a distribution over $\{\pm1\}^n$ in the subcube conditional model. Our $\widetilde{O}(n^{3/2}/\varepsilon^2)$-query algorithm for unateness testing will rely on their principal technical lemma, which we state after introducing some relevant notation.

For any $x \in \{\pm 1\}^n$ and $i \in [n]$, we define

$$\delta_{x,i} \coloneqq \frac{\mathcal{D}(x^{i\to -1}) - \mathcal{D}(x^{i\to +1})}{\mathcal{D}(x^{i\to -1}) + \mathcal{D}(x^{i\to +1})} \in [-1,1]$$

as the bias on the edge $\{x, x^{\oplus i}\}$. Note that if $\mathcal{D}$ is monotone, we always have $\delta_{x,i} \le 0$, so a positive $\delta_{x,i}$ witnesses a violation of monotonicity.

Central to the analysis in [21] is the following lemma, which is implicit in the proof of [21, Lemma 2.3]. For $x \in \mathbb{R}$, we write $(x)_+ \coloneqq \max\{x, 0\}$.

▶ **Lemma 14** ([21]). *For any $\mathcal{D}$ that is ε-far from monotone, there exists a positive integer $w \le O(\log(n/\varepsilon))$ such that*

$$\mathop{\mathbf{Pr}}_{\substack{\boldsymbol{x}\sim\mathcal{D}\\ \boldsymbol{i}\sim[n]}}\left[(\delta_{\boldsymbol{x},\boldsymbol{i}})_+^2 \ge 2^{-w}\right] \ge \widetilde{\Omega}\left(\frac{2^w\cdot\varepsilon^2}{n}\right).$$

The proof of Lemma 14 relies on a real-valued "directed" isoperimetric inequality proved in [21]. In particular, it follows from Lemma 14 that if we draw a random $\boldsymbol{w} \le O(\log(n/\varepsilon))$ with probability $\boldsymbol{w} = w$ proportional to 2^{-w}, we have

$$\mathop{\mathbf{Pr}}_{\boldsymbol{w},\boldsymbol{x}\sim\mathcal{D},\boldsymbol{i}\sim[n]}\left[(\delta_{\boldsymbol{x},\boldsymbol{i}})_+^2 \ge 2^{-\boldsymbol{w}}\right] \ge \sum_{w=1}^{O(\log(n/\varepsilon))} 2^{-w}\cdot \mathop{\mathbf{Pr}}_{\substack{\boldsymbol{x}\sim\mathcal{D}\\ \boldsymbol{i}\sim[n]}}\left[(\delta_{\boldsymbol{x},\boldsymbol{i}})_+^2 \ge 2^{-w}\right] \ge \widetilde{\Omega}\left(\frac{\varepsilon^2}{n}\right). \tag{16}$$

Give the lemma above, the algorithm of [21] is natural: we simply sample $\widetilde{O}(n/\varepsilon^2)$ triples $(\boldsymbol{w},\boldsymbol{x},\boldsymbol{i})$, in the hope of finding at least one with $(\delta_{\boldsymbol{x},\boldsymbol{i}})_+^2 \ge 2^{-\boldsymbol{w}}$. For each triple, we make $\widetilde{O}(2^w)$ queries to the subcube (or edge) $\{x, x^{\oplus i}\}$ to check whether $(\delta_{x,i})_+^2 \ge 2^{-w}$ holds. If this holds for any triple, we reject distribution $\mathcal{D}$. Over the randomness in $\boldsymbol{w}$, we make $\sum_{w=1}^{O(\log(n/\varepsilon))} 2^{-w}\cdot\widetilde{O}(2^w) = \widetilde{O}(1)$ queries for each triple, so the overall query complexity is $\widetilde{O}(n/\varepsilon^2)$.

4.1.2 Proof of Theorem 5

We first introduce some additional notation before proving Theorem 5. Recall from Notation 7 that, for $\sigma \in \{\pm 1\}^n$, $\mathcal{D}^\sigma$ is the distribution of

$$\boldsymbol{x}^\sigma \coloneqq (\sigma_1\boldsymbol{x}_1, \sigma_2\boldsymbol{x}_2, \ldots, \sigma_n\boldsymbol{x}_n),$$

where $\boldsymbol{x} \sim \mathcal{D}$, i.e., $\mathcal{D}^\sigma$ is obtained from $\mathcal{D}$ by flipping the coordinates indexed by $\{i \in [n] : \sigma_i = -1\}$. Recall that, with respect to distribution $\mathcal{D}$, we defined

$$\delta_{x,i} \coloneqq \frac{\mathcal{D}(x^{i\to -1}) - \mathcal{D}(x^{i\to +1})}{\mathcal{D}(x^{i\to -1}) + \mathcal{D}(x^{i\to +1})}.$$

In the following, we will write $\delta_{x,i}^{\mathcal{D}}$ to avoid confusion and emphasize dependence on $\mathcal{D}$. While the next identity is notation-heavy, it states an intuitive fact: if $\mathcal{D}^\sigma$ has a bias on an edge adjacent to x along the ith direction, $\mathcal{D}$ should also have a bias at x^σ, albeit flipped by σ_i. More formally,

$$\begin{aligned}
\delta_{x,i}^{\mathcal{D}^\sigma} &= \frac{\mathcal{D}^\sigma(x^{i\to -1}) - \mathcal{D}^\sigma(x^{i\to +1})}{\mathcal{D}^\sigma(x^{i\to -1}) + \mathcal{D}^\sigma(x^{i\to +1})} \\
&= \frac{\mathcal{D}((x^{i\to -1})^\sigma) - \mathcal{D}((x^{i\to +1})^\sigma)}{\mathcal{D}((x^{i\to -1})^\sigma) + \mathcal{D}((x^{i\to +1})^\sigma)} && (\mathcal{D}^\sigma(x) = \mathcal{D}(x^\sigma)) \\
&= \frac{\sigma_i\cdot\left[\mathcal{D}((x^\sigma)^{i\to -1}) - \mathcal{D}((x^\sigma)^{i\to +1})\right]}{\mathcal{D}((x^\sigma)^{i\to -1}) + \mathcal{D}((x^\sigma)^{i\to +1})} \\
&= \sigma_i\cdot\delta_{x^\sigma,i}^{P}. && (17)
\end{aligned}$$

We now record a corollary of Lemma 14 which will be crucial to our analysis of Algorithm 2. Since $\mathcal{D}$ is ε-far from unate, for every sign pattern $\sigma \in \{\pm 1\}^n$, the distribution $\mathcal{D}^\sigma$ is ε-far from monotone. Applying Lemma 14 (or more precisely, Equation (16)) to $\mathcal{D}^\sigma$ then gives

$$\begin{aligned}\widetilde{\Omega}\left(\frac{\varepsilon^2}{n}\right) &\leq \mathop{\mathbf{Pr}}_{\boldsymbol{w},\boldsymbol{x}\sim\mathcal{D}^\sigma,\boldsymbol{i}\sim[n]}\left[(\delta^{\mathcal{D}^\sigma}_{\boldsymbol{x},\boldsymbol{i}})^2_+ \geq 2^{-\boldsymbol{w}}\right] \\ &= \mathop{\mathbf{Pr}}_{\boldsymbol{w},\boldsymbol{x}\sim\mathcal{D}^\sigma,\boldsymbol{i}\sim[n]}\left[(\sigma_i \cdot \delta^{\mathcal{D}}_{\boldsymbol{x}^\sigma,\boldsymbol{i}})^2_+ \geq 2^{-\boldsymbol{w}}\right] && \text{(Equation (17))} \\ &= \mathop{\mathbf{Pr}}_{\boldsymbol{w},\boldsymbol{x}\sim\mathcal{D},\boldsymbol{i}\sim[n]}\left[(\sigma_i \cdot \delta^{\mathcal{D}}_{\boldsymbol{x},\boldsymbol{i}})^2_+ \geq 2^{-\boldsymbol{w}}\right], && (18)\end{aligned}$$

where Equation (18) relies on the fact that for $\boldsymbol{x} \sim \mathcal{D}^\sigma$, the random variable $\boldsymbol{x}^\sigma$ is distributed as $\mathcal{D}$. We record this consequence of Lemma 14 for future use:

▶ **Corollary 15.** *Suppose $\mathcal{D}$ is ε-far from unate. Then for every $\sigma \in \{\pm 1\}^n$, we have*

$$\mathop{\mathbf{Pr}}_{\boldsymbol{w},\boldsymbol{x}\sim\mathcal{D},\boldsymbol{i}\sim[n]}\left[(\sigma_i \cdot \delta^{\mathcal{D}}_{\boldsymbol{x},\boldsymbol{i}})^2_+ \geq 2^{-\boldsymbol{w}}\right] \geq \widetilde{\Omega}\left(\frac{\varepsilon^2}{n}\right).$$

Corollary 15 suggests a natural test to detect distributions that are far from unate and forms the basis of Algorithm 2: iterating over each coordinate $i \in [n]$, we reject if there exist two pairs $(\boldsymbol{w}, \boldsymbol{x})$ and $(\boldsymbol{w}', \boldsymbol{x}')$ such that

$$(\delta^{\mathcal{D}}_{\boldsymbol{x},i})^2_+ \geq 2^{-\boldsymbol{w}} \quad \text{and} \quad (-\delta^{\mathcal{D}}_{\boldsymbol{x}',i})^2_+ \geq 2^{-\boldsymbol{w}'}. \tag{19}$$

Intuitively, the former condition rules out the possibility that $\mathcal{D}$ is monotone in direction i, and the latter rules out that $\mathcal{D}$ is anti-monotone in direction i. Note that if $\mathcal{D}$ is unate, then both conditions above can never hold simultaneously.

We now turn to the proof of Theorem 5:

Proof of Theorem 5. For a fixed triple $(i, \boldsymbol{x}, \boldsymbol{w})$, the expected number of conditional samples is

$$\sum_{t=1}^{\Theta(\log(n/\varepsilon))} \mathbf{Pr}[\boldsymbol{w} = t] \cdot T(t) = \sum_t \Theta(2^{-t}) \cdot \Theta(2^t \log(nk)) = \widetilde{O}(1).$$

We perform $\Theta(n \cdot k) = \widetilde{\Theta}(n^{3/2}/\varepsilon^2)$ triples, so the total query complexity is $\widetilde{O}(n^{3/2}/\varepsilon^2)$.

By Hoeffding's inequality,

$$\mathbf{Pr}\left[\left|\boldsymbol{\tau}_\ell - \delta^{\mathcal{D}}_{\boldsymbol{x},i}\right| \;\geq\; \frac{1}{8}\,2^{-\boldsymbol{w}/2}\right] \;\leq\; 2\exp\Big(-\tfrac{1}{2}\cdot(\frac{1}{8}\,2^{-\boldsymbol{w}/2})^2\,T(\boldsymbol{w})\Big) \;\leq\; \frac{\delta_{\mathrm{cmp}}}{6nk},$$

for a sufficiently large absolute constant in the definition of $T(\boldsymbol{w})$. Taking a union bound over all nk trials, with probability at least $1 - \delta_{\mathrm{cmp}}/6$ we have, simultaneously for all trials,

$$\boldsymbol{\tau}_\ell \in \Big[\delta^{\mathcal{D}}_{\boldsymbol{x},i} - \frac{1}{8}\,2^{-\boldsymbol{w}/2},\; \delta^{\mathcal{D}}_{\boldsymbol{x},i} + \frac{1}{8}\,2^{-\boldsymbol{w}/2}\Big]. \tag{20}$$

For the remainder of the argument, we condition on the event that Equation (20) holds for all trials.

Algorithm 2 The `Subcube-Unate` algorithm.

Input: Subcube query access to $\mathcal{D}$, distance parameter $\varepsilon \in (0,1)$

Output: "Accept" or "reject"

`Subcube-Unate`$(\mathcal{D}, \varepsilon)$:

1. Set $k := \widetilde{\Theta}\left(\frac{\sqrt{n}}{\varepsilon^2}\right)$ and repeat Step 2 $\Theta(1)$-times:
2. For $i \in [n]$:
   a. For $\ell \in [k]$:
      i. Sample $\boldsymbol{x} \sim \mathcal{D}$ and consider the restriction $\boldsymbol{\rho} \in \{\pm 1, *\}^n$ given by $\boldsymbol{\rho}_j = \boldsymbol{x}_j$ if $j \neq i$ and $\boldsymbol{\rho}_i = *$.
      ii. Draw $\boldsymbol{w} \in [\Theta(\log(n/\varepsilon))]$ where $\mathbf{Pr}[\boldsymbol{w} = t] \propto 2^{-t}$.
      iii. Query the conditional distribution $\mathcal{D}\,|_{\boldsymbol{\rho}}$ independently

$$T(\boldsymbol{w}) := \Theta(1) \cdot 2^{\boldsymbol{w}} \cdot \ln(12nk/\delta_{\mathrm{cmp}})$$

      times for $\delta_{\mathrm{cmp}} := n^{-3}$. Let $N_+^{(\ell)}$ and $N_-^{(\ell)}$ be the numbers of returns with coordinate i equal to $+1$ and -1, respectively ($\boldsymbol{N}_+^{(\ell)} + \boldsymbol{N}_-^{(\ell)} = T(\boldsymbol{w})$). Define the empirical edge bias

$$\boldsymbol{\tau}_\ell \;:=\; \frac{\boldsymbol{N}_-^{(\ell)} - \boldsymbol{N}_+^{(\ell)}}{\boldsymbol{N}_-^{(\ell)} + \boldsymbol{N}_+^{(\ell)}}\,, \qquad \text{and set} \;\; c_{\boldsymbol{w}} := \frac{2^{-\boldsymbol{w}/2}}{4}\,.$$

      Mark the trial ℓ as a *positive witness* if $\boldsymbol{\tau}_\ell \geq c_{\boldsymbol{w}}$, and as a *negative witness* if $\boldsymbol{\tau}_\ell < -c_{\boldsymbol{w}}$.
   b. If there exist $\ell_1, \ell_2 \in [k]$ such that ℓ_1 is a positive witness and ℓ_2 is a negative witness, then halt and output "reject."
3. If the algorithm has not rejected, output "accept."

Completeness. Conditioned on the above event, it is readily seen that `Subcube-Unate` has one-sided error. In particular, suppose $\mathcal{D}$ is a unate distribution. It follows that there exists $\sigma \in \{\pm 1\}^n$ such that $\mathcal{D}^\sigma$ is monotone. In particular, $\sigma_i \cdot \delta^{\mathcal{D}}_{x,i} \leq 0$ for all $x \in \{\pm 1\}^n$ and $i \in [n]$. Fix a coordinate $i \in [n]$ and suppose $\sigma_i = +1$. We have $\delta^{\mathcal{D}}_{\boldsymbol{x},i} \leq 0$ for every $\boldsymbol{x}$, and conditioned on Equation (20),

$$\boldsymbol{\tau}_\ell \leq \; \tfrac{1}{8}\, 2^{-\boldsymbol{w}/2} \;<\; c_{\boldsymbol{w}} = \frac{2^{-\boldsymbol{w}/2}}{4}\,,$$

so this trial cannot create a positive witness for coordinate i. An identical argument gives that if $\sigma_i = -1$, then the trial cannot create a negative witness for coordinate i. It follows that, conditioned on Equation (20), the algorithm will output "accept" with probability 1.

Soundness. We now turn to soundness of `Subcube-Unate`$(\mathcal{D}, \varepsilon)$. Suppose $\mathcal{D}$ is ε-far from every unate distribution, we will show that `Subcube-Unate`$(\mathcal{D}, \varepsilon)$ will output "reject" with probability at least 9/10. For $\boldsymbol{w}$ distributed as in Step 2(a).ii of Algorithm 2 and $i \in [n]$, let

$$q_i^+ := \mathop{\mathbf{Pr}}_{\boldsymbol{w},\boldsymbol{x}\sim\mathcal{D}}\left[(\delta^{\mathcal{D}}_{\boldsymbol{x},i})_+^2 \geq 2^{-\boldsymbol{w}}\right] \qquad \text{and} \qquad q_i^- := \mathop{\mathbf{Pr}}_{\boldsymbol{w},\boldsymbol{x}\sim\mathcal{D}}\left[(-\delta^{\mathcal{D}}_{\boldsymbol{x},i})_+^2 \geq 2^{-\boldsymbol{w}}\right]$$

be the probabilities that, over the randomness in $\boldsymbol{w}$ and $\boldsymbol{x}$, the pair $(\boldsymbol{w}, \boldsymbol{x})$ witnesses that the ith coordinate cannot be monotone and anti-monotone respectively. Corollary 15 directly implies that

$$m := \frac{1}{n}\sum_{i=1}^{n} m_i \geq \widetilde{\Omega}\left(\frac{\varepsilon^2}{n}\right), \qquad \text{where } m_i := \min\left\{q_i^+, q_i^-\right\}. \tag{21}$$

Let r_i be the probability that $\texttt{Subcube-Unate}(\mathcal{D}, \varepsilon)$ outputs "reject" in Step 2(b) on coordinate i. Thanks to independence, we have

$$\mathbf{Pr}\left[\texttt{Subcube-Unate}(\mathcal{D}, \varepsilon) \text{ outputs "accept"}\right] = \prod_{i=1}^{n}(1 - r_i) \leq \exp\left(-\sum_{i=1}^{n} r_i\right),$$

and so the remainder of the argument will establish that $\sum_{i=1}^{n} r_i = \Omega(1)$.

First, note that

$$r_i \geq \left(1 - (1 - q_i^+)^{k/2}\right)\left(1 - (1 - q_i^-)^{k/2}\right) \geq \left(1 - (1 - m_i)^{k/2}\right)^2, \tag{22}$$

where the first inequality holds because the probability of returning "reject" after k iterations of Steps 2(a).i–iii is at least the probability of detecting a positive witness among the first $k/2$ iterations and a negative witness among the last $k/2$ iterations. The following lemma will be useful:

▶ **Lemma 16.** *For $K \geq 1$, the function $f(x) = (1 - (1 - x)^K)^2$ is convex on the interval $(0, \ln(2)/K)$.*

Proof. We explicitly compute the first and second derivatives of the function f. Note that

$$f'(x) = 2K\Big(1 - (1 - x)^K\Big)(1 - x)^{K-1}.$$

Taking another derivative gives

$$\begin{aligned} f''(x) &= 2K^2(1 - x)^{2K-2} - 2K(K - 1)(1 - (1 - x)^K)(1 - x)^{K-2} \\ &= 2K(1 - x)^{K-2}((2K - 1)(1 - x)^K - (K - 1)). \end{aligned}$$

So it suffices to show

$$(2K - 1)\Big(1 - \tfrac{\ln 2}{K}\Big)^K \geq K - 1,$$

or equivalently $F(K) \geq 0$ where $F(x) := x\ln\Big(1 - \frac{\ln 2}{x}\Big) - \ln\Big(\frac{x-1}{2x-1}\Big)$. Since

$$\lim_{x \to +\infty} F(x) = -\ln 2 - \ln\tfrac{1}{2} = 0,$$

showing that $F'(x) < 0$ for all $x \in (1, +\infty)$ will complete the proof.

Note that

$$F'(x) = \ln\Big(1 - \frac{1}{x/\ln 2}\Big) + \frac{1}{x/\ln 2 - 1} - \frac{1}{(x-1)(2x-1)}.$$

Furthermore, the function $h(y) := \ln\Big(1 - \frac{1}{y}\Big) + \frac{1}{y-1}$ has derivative $h'(y) = -\frac{1}{y(y-1)^2}$, and is thus strictly decreasing on $(1, +\infty)$. If we could show that

$$h(x) = \ln\Big(1 - \frac{1}{x}\Big) + \frac{1}{x-1} \leq \frac{1}{(x-1)(2x-1)} \tag{23}$$

holds for all $x \in (1, +\infty)$, we would have the desired inequality

$$F'(x) = h(x/\ln 2) - \frac{1}{(x-1)(2x-1)} < h(x) - \frac{1}{(x-1)(2x-1)} \leq 0.$$

Equation (23) follows immediately from the well-known/readily verified inequality $2a \leq (2+a) \cdot \ln(1+a)$ for $a \geq 0$ by taking $a = (x-1)^{-1}$. ◀

Returning to the soundness of `Subcube-Unate`, note that if there exists a coordinate j with $m_j \geq \frac{2\ln 2}{k}$, then thanks to Equation (22) we have

$$r_j \geq (1-(1-m_j)^{k/2})^2 \geq \left(1 - \left(1 - \frac{\ln 2}{k/2}\right)^{k/2}\right)^2 \geq (1 - e^{-\ln 2})^2 = \frac{1}{4}.$$

Consequently, the amplification in Step 1 ensures that one of the $\Theta(1)$ independent runs will output "reject" with probability at least $9/10$.

On the other hand, suppose $m_i < \frac{2\ln 2}{k}$ for all $i \in [n]$. Thanks to Lemma 16 and Jensen's inequality, we then have

$$\sum_{i=1}^{n} r_i \geq \sum_{i=1}^{n} \left(1 - (1-m_i)^{k/2}\right)^2 \geq n\left(1 - (1-m)^{k/2}\right)^2.$$

It follows that $\sum_{i=1}^{n} r_i \geq 99/100$ from Equation (21) and our choice of k by an easy application of Bernoulli's inequality $(1+u)^r \geq 1 + ru$ for $r, u \geq 0$. ◀

4.2 Lower Bound

We prove the following lower bound for testing unateness of distributions in the subcube model, which applies to all testers that only make subcube queries on either the entire hypercube (to get an unconditional sample) or a small subcube.

▶ **Theorem 6.** *There exists a constant $\varepsilon_0 > 0$ such that the following holds: If an algorithm tests whether an unknown distribution $\mathcal{D}$ over $\{\pm 1\}^n$ is unate or ε_0-far from unate by drawing Q_1 samples from $\mathcal{D}$ and making Q_2 (possibly adaptive) subcube queries on subcubes of dimensions $\leq d$, we have*

$$Q_1 + 2^d Q_2 \geq \Omega(n^{2/3}/\log^3 n).$$

Recall that our tester for Theorem 5, like the monotonicity tester of [21], only makes subcube queries on edges (i.e., one-dimensional subcubes) in addition to drawing unconditional samples. In other words, our tester satisfies the precondition of Theorem 6 with $d = 1$. The theorem then implies that all such testers must make at least $\widetilde{\Omega}(n^{2/3})$ queries in total.

The proof is deferred to the full version [45].

References

1 Jayadev Acharya, Constantinos Daskalakis, and Gautam Kamath. Optimal testing for properties of distributions. *Advances in Neural Information Processing Systems*, 28, 2015.

2 Michał Adamaszek, Artur Czumaj, and Christian Sohler. Testing monotone continuous distributions on high-dimensional real cubes. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 56–65. SIAM, 2010.

3 Tomer Adar, Eldar Fischer, and Amit Levi. Improved bounds for high-dimensional equivalence and product testing using subcube queries. *arXiv preprint*, 2024. doi:10.48550/arXiv.2408.02347.

4 Maryam Aliakbarpour, Themis Gouleakis, John Peebles, Ronitt Rubinfeld, and Anak Yodpinyanee. Towards testing monotonicity of distributions over general posets. In *Conference on Learning Theory*, pages 34–82. PMLR, 2019. URL: http://proceedings.mlr.press/v99/aliakbarpour19a.html.

5 Roksana Baleshzar, Deeparnab Chakrabarty, Ramesh Krishnan S Pallavoor, Sofya Raskhodnikova, and C Seshadhri. Optimal unateness testers for real-valued functions: Adaptivity helps. *Theory of Computing*, 16(1):1–36, 2020. doi:10.4086/TOC.2020.V016A003.

6 Tugkan Batu, Ravi Kumar, and Ronitt Rubinfeld. Sublinear algorithms for testing monotone and unimodal distributions. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 381–390, 2004. doi:10.1145/1007352.1007414.

7 V. Bentkus. On the dependence of the Berry-Esseen bound on dimension. *Journal of Statistical Planning and Inference*, 113:385–402, 2003.

8 Ivona Bezáková, Antonio Blanca, Zongchen Chen, Daniel Štefankovič, and Eric Vigoda. Lower bounds for testing graphical models: Colorings and antiferromagnetic ising models. *Journal of Machine Learning Research*, 21(25):1–62, 2020. URL: https://jmlr.org/papers/v21/19-580.html.

9 Arnab Bhattacharyya, Eldar Fischer, Ronitt Rubinfeld, and Paul Valiant. Testing monotonicity of distributions over general partial orders. In *ICS*, pages 239–252, 2011. URL: http://conference.iiis.tsinghua.edu.cn/ICS2011/content/papers/38.html.

10 Rishiraj Bhattacharyya and Sourav Chakraborty. Property testing of joint distributions using conditional samples. *ACM Transactions on Computation Theory (TOCT)*, 10(4):1–20, 2018. doi:10.1145/3241377.

11 Hadley Black, Deeparnab Chakrabarty, and C Seshadhri. A $d^{1/2+o(1)}$ monotonicity tester for boolean functions on d-dimensional hypergrids. *SIAM Journal on Computing*, pages FOCS23–147, 2025.

12 Guy Blanc, Jane Lange, Ali Malik, and Li-Yang Tan. Lifting uniform learners via distributional decomposition. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1755–1767, 2023. doi:10.1145/3564246.3585212.

13 Antonio Blanca, Zongchen Chen, Daniel Štefankovič, and Eric Vigoda. Complexity of high-dimensional identity testing with coordinate conditional sampling. *ACM Transactions on Algorithms*, 21(1):1–58, 2024. doi:10.1145/3686799.

14 N. Bshouty and C. Tamon. On the Fourier spectrum of monotone functions. *Journal of the ACM*, 43(4):747–770, 1996. doi:10.1145/234533.234564.

15 Clément L Canonne. A survey on distribution testing: Your data is big. but is it blue? *Theory of Computing*, pages 1–100, 2020.

16 Clément L Canonne, Xi Chen, Gautam Kamath, Amit Levi, and Erik Waingarten. Random restrictions of high dimensional distributions and uniformity testing with subcube conditioning. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 321–336. SIAM, 2021. doi:10.1137/1.9781611976465.21.

17 Clément L Canonne, Ilias Diakonikolas, Daniel M Kane, and Alistair Stewart. Testing bayesian networks. In *Conference on Learning Theory*, pages 370–448. PMLR, 2017. URL: http://proceedings.mlr.press/v65/canonne17a.html.

18 Clément L Canonne et al. Topics and techniques in distribution testing: A biased but representative sample. *Foundations and Trends® in Communications and Information Theory*, 19(6):1032–1198, 2022. doi:10.1561/0100000114.

19 Clément L Canonne, Elena Grigorescu, Siyao Guo, Akash Kumar, and Karl Wimmer. Testing k-monotonicity: The rise and fall of boolean functions. *Theory of Computing*, 15(1):1–55, 2019. doi:10.4086/TOC.2019.V015A001.

20 Clément L Canonne, Dana Ron, and Rocco A Servedio. Testing probability distributions using conditional samples. *SIAM Journal on Computing*, 44(3):540–616, 2015. doi:10.1137/130945508.

21 Deeparnab Chakrabarty, Xi Chen, Simeon Ristic, C. Seshadhri, and Erik Waingarten. Monotonicity testing of high-dimensional distributions with subcube conditioning. In *Symposium on Theory of Computing (STOC)*, pages 1019–1030, 2025. doi:10.1145/3717823.3718297.

22 Deeparnab Chakrabarty and C Seshadhri. A $o(n)$ monotonicity tester for boolean functions over the hypercube. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, pages 411–418, 2013. doi:10.1145/2488608.2488660.

23 Deeparnab Chakrabarty and C Seshadhri. A $O(n)$ non-adaptive tester for unateness. *arXiv preprint*, 2016. arXiv:1608.06980.

24 Sourav Chakraborty, Eldar Fischer, Yonatan Goldhirsh, and Arie Matsliah. On the power of conditional samples in distribution testing. *SIAM Journal on Computing*, 45(4):1261–1296, 2016. doi:10.1137/140964199.

25 Mark Chen, Xi Chen, Hao Cui, William Pires, and Jonah Stockwell. Boolean function monotonicity testing requires (almost) $n^{1/2}$ queries. *arXiv preprint*, 2025. doi:10.48550/arXiv.2511.04558.

26 Xi Chen, Anindya De, Yuhao Li, Shivam Nadimpalli, and Rocco A Servedio. Mildly exponential lower bounds on tolerant testers for monotonicity, unateness, and juntas. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 4321–4337. SIAM, 2024. doi:10.1137/1.9781611977912.151.

27 Xi Chen, Rajesh Jayaram, Amit Levi, and Erik Waingarten. Learning and testing junta distributions with sub cube conditioning. In *Conference on Learning Theory*, pages 1060–1113. PMLR, 2021. URL: http://proceedings.mlr.press/v134/chen21b.html.

28 Xi Chen and Cassandra Marcussen. Uniformity testing over hypergrids with subcube conditioning. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 4338–4370. SIAM, 2024. doi:10.1137/1.9781611977912.152.

29 Xi Chen, Rocco A Servedio, and Li-Yang Tan. New algorithms and lower bounds for monotonicity testing. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 286–295. IEEE, 2014. doi:10.1109/FOCS.2014.38.

30 Xi Chen and Erik Waingarten. Testing unateness nearly optimally. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 547–558, 2019. doi:10.1145/3313276.3316351.

31 Xi Chen, Erik Waingarten, and Jinyu Xie. Beyond talagrand functions: new lower bounds for testing monotonicity and unateness. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 523–536, 2017. doi:10.1145/3055399.3055461.

32 Xi Chen, Erik Waingarten, and Jinyu Xie. Boolean unateness testing with $\tilde{O}(n^{3/4})$ adaptive queries. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 868–879, 2017. doi:10.1109/FOCS.2017.85.

33 Zhihuai Chen, Siyao Guo, Qian Li, Chengyu Lin, and Xiaoming Sun. New distinguishers for negation-limited weak pseudorandom functions. *arXiv preprint*, 2022. doi:10.48550/arXiv.2203.12246.

34 Constantinos Daskalakis, Nishanth Dikkala, and Gautam Kamath. Testing ising models. *IEEE Transactions on Information Theory*, 65(11):6829–6852, 2019. doi:10.1109/TIT.2019.2932255.

35 Constantinos Daskalakis and Qinxuan Pan. Square hellinger subadditivity for bayesian networks and its applications to identity testing. In *Conference on Learning Theory*, pages 697–703. PMLR, 2017. URL: http://proceedings.mlr.press/v65/daskalakis17a.html.

36 Anindya De, Huan Li, Shivam Nadimpalli, and Rocco A Servedio. Detecting low-degree truncation. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1027–1038, 2024. doi:10.1145/3618260.3649633.

37 Anindya De, Shivam Nadimpalli, and Rocco A. Servedio. Testing convex truncation. *Mathematical Statistics and Learning*, 2025. Preliminary version in *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. doi:10.4171/MSL/50.

38 O. Goldreich, S. Goldwasser, E. Lehman, D. Ron, and A. Samordinsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000. doi:10.1007/S004930070011.

39 Elena Grigorescu, Akash Kumar, and Karl Wimmer. Flipping out with many flips: Hardness of testing k-monotonicity. *SIAM Journal on Discrete Mathematics*, 33(4):2111–2125, 2019. doi:10.1137/18M1217978.

40 William He and Shivam Nadimpalli. Testing junta truncation. *arXiv preprint*, 2023. doi:10.48550/arXiv.2308.13992.

41 Subhash Khot, Dor Minzer, and Muli Safra. On monotonicity testing and boolean isoperimetric-type theorems. *SIAM Journal on Computing*, 47(6):2238–2276, 2018. doi:10.1137/16M1065872.

42 Subhash Khot and Igor Shinkar. An $\tilde{O}(n)$ Queries Adaptive Tester for Unateness. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2016)*, volume 60 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 37:1–37:7. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.APPROX-RANDOM.2016.37.

43 Jane Lange, Ronitt Rubinfeld, and Arsen Vasilyan. Properly learning monotone functions via local correction. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 75–86. IEEE, 2022. doi:10.1109/FOCS54457.2022.00015.

44 Jane Lange and Arsen Vasilyan. Agnostic proper learning of monotone functions: beyond the black-box correction barrier. *SIAM Journal on Computing*, pages FOCS23–1, 2025.

45 Daeho Lee, Shivam Nadimpalli, Mingda Qiao, and Ronitt Rubinfeld. Testing unate distributions, 2026. arXiv:2607.01573.

46 Kuldeep S Meel, Yash Pralhad Pote, and Sourav Chakraborty. On testing of samplers. *Advances in Neural Information Processing Systems*, 33:5753–5763, 2020.

47 Shyam Narayanan. On tolerant distribution testing in the conditional sampling model. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 357–373. SIAM, 2021. doi:10.1137/1.9781611976465.23.

48 Donald B Owen. Tables for computing bivariate normal probabilities. *The Annals of Mathematical Statistics*, 27(4):1075–1090, 1956.

49 Yash Pote and Kuldeep S Meel. On scalable testing of samplers. *Advances in Neural Information Processing Systems*, 35:28068–28079, 2022.

50 Mingda Qiao and Gregory Valiant. Learning Discrete Distributions from Untrusted Batches. In *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*, volume 94 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 47:1–47:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ITCS.2018.47.

51 Ronitt Rubinfeld and Rocco A. Servedio. Testing monotone high-dimensional distributions. *Random Struct. Algorithms*, 34(1):24–44, 2009. doi:10.1002/RSA.20247.

52 Ronitt Rubinfeld and Arsen Vasilyan. Monotone probability distributions over the boolean cube can be learned with sublinear samples. *11th Innovations in Theoretical Computer Science (ITCS 2020)*, 2020. doi:10.4230/LIPIcs.ITCS.2020.28.

53 M. Talagrand. Isoperimetry, logarithmic Sobolev inequalities on the discrete cube and Margulis' graph connectivity theorem. *GAFA*, 3(3):298–314, 1993.

Homomorphism Testing with Resilience to Online Manipulations

Esty Kelman
Massachusetts Institute of Technology, Cambridge, MA, USA
Boston University, MA, USA

Uri Meir
Blavatnik School of Computer Science, Tel Aviv University, Israel

Debanuj Nayak
Boston University, MA, USA

Sofya Raskhodnikova
Boston University, MA, USA

Abstract

A central challenge in property testing is verifying algebraic structure with minimal access to data. A landmark result addressing this challenge, the linearity test of Blum, Luby, and Rubinfeld (JCSS '93), spurred a rich body of work on testing algebraic properties such as linearity and its generalizations to low-degree polynomials and group homomorphisms. However, classical tests for these properties assume unrestricted, noise-free access to the input function – an assumption that breaks down in adversarial or dynamic settings. To address this, Kalemaj, Raskhodnikova, and Varma (Theory of Computing '23) introduced the online manipulation model, where an adversary erases or corrupts query responses over time, based on the tester's past queries.

We initiate the study of *manipulation-resilient* testing for *group homomorphism* in this online model. Our main result is an *optimal* tester that makes $O(1/\varepsilon + \log t)$ queries, where ε is the distance parameter and t is the number of function values the adversary can erase or corrupt per query. Our result recovers the celebrated $O(1/\varepsilon)$ bound by Ben-Or, Coppersmith, Luby, and Rubinfeld (Random Struct. Algorithms '08) for homomorphism testing in the standard property testing model, albeit with a different tester. Our tester, Random Signs Test, *lifts* known manipulation-resilient linearity testers for $\mathbb{F}_2^n \to \mathbb{F}_2$ to general group domains and codomains by introducing more randomness: instead of verifying the homomorphism condition for a sum of random elements, it uses additions and subtractions of random elements, randomly selecting a sign for each element. We also obtain improved group-specific query bounds for key families of groups. Our results show that despite the challenges of online manipulation, group homomorphism – a fundamental algebraic property – is efficiently testable across a wide range of domains and codomains. Finally, we formalize a general framework for proving online resilience.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases Property Testing, Sublinear Algorithms, Online Manipulation Resilience, Group Theory

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.58

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2511.23363` [35]

Funding *Esty Kelman*: Partially supported by NSF awards DMS-2022448, DMS-2022446, and CCF-2227876.
Debanuj Nayak: Supported in part by NSF award 2022446.
Sofya Raskhodnikova: Supported in part by NSF award 2022446.

Acknowledgements The authors would like to thank Ephraim Linder for useful discussions. Part of this research was carried out during the authors' visit to the Simons Institute for the Theory of Computing at UCB.



© Esty Kelman, Uri Meir, Debanuj Nayak, and Sofya Raskhodnikova;
licensed under Creative Commons License CC-BY 4.0



1 Introduction

Understanding how to verify algebraic structure with minimal access to data is a central question in *property testing*. A landmark result addressing this challenge is the linearity test of Blum, Luby, and Rubinfeld [15], which checks whether a given function is linear by querying it on two random inputs and their sum. This test and its variants sparked a rich line of work on testing algebraic properties of functions, including linearity [6, 21, 8], being a low-degree polynomial [5, 23, 22, 53, 51, 1, 3, 46, 33, 54, 56, 30, 14, 28, 52, 17, 32], and being a group homomorphism [15, 27, 11, 25, 7, 45, 26, 43], with deep connections to probabilistically-checkable proofs [2, 59, 29].

However, classical property testing, as defined by [53, 24], relies on the assumption of undisrupted query access to the input function – an assumption that may break down in dynamic environments or under adversarial or incomplete access to data. Many testers, especially for algebraic properties, use structured queries that can be vulnerable to online adversarial interference. To address this, Kalemaj, Raskhodnikova, and Varma [31] introduced the *online manipulation model,* where an adversary can erase or corrupt the input during the execution of the algorithm, based on prior queries. This framework has prompted the reexamination of classical testing problems under a new lens, seeking algorithms that remain correct and efficient despite adversarial interference.

In this work, we initiate the study of manipulation-resilient testing for group homomorphism, a fundamental property of functions between algebraic structures. Given finite groups $(G, +)$ and $(H, \oplus)$, and a function $f : G \to H$, the goal is to determine whether f is a *homomorphism*[1] – i.e., satisfies $f(x_1 + x_2) = f(x_1) \oplus f(x_2)$ for all $x_1, x_2 \in G$ – or is ε-far from every homomorphism.[2] The tester accesses the input function f by querying its values $f(x)$ on elements x of the group, but must contend with an *online adversary* that knows all queries made by the tester and can manipulate the data after answering each query. Two types of manipulation are considered: an *erasure*, where $f(x)$ is replaced by a special symbol $\perp$, and a *corruption*, where $f(x)$ is replaced by an arbitrary element of H. The rate of manipulations is controlled by a parameter t. Two types of adversaries have been studied: *fixed rate*, which can manipulate t values after every query is answered [31], and *budget-managing*, which accumulates a budget of t manipulations per query and may deploy them later at any point in the computation, after answering any query [10].

Linearity and low-degree testing are well understood in the online adversarial model. Linearity and quadraticity were among the first properties studied in this setting, with [31] proving a query lower bound of $\Omega(\log t)$ for testing linearity of functions $f : \mathbb{F}_2^n \to \mathbb{F}_2$. fThe linearity tester of [31] was improved by Ben-Eliezer, Kelman, Meir and Raskhodnikova [10], who achieved the optimal query complexity of $O(1/\varepsilon + \log t)$, and further refined by Arora, Kelman, and Meir [4] to handle the full range of the parameter t. For the related problem of direct sum testing, Westover, Yu and Zheng [60] achieve the same query complexity. For testing whether a function $f : \mathbb{F}_q^n \to \mathbb{F}_q$ has degree at most d, for fixed constants d, q and ε, a lower bound of $\Omega(\log^d t)$ on the query complexity was first shown by [10] for $q = 2$ and later by [36] whenever $q \geq d$, whereas the elegant tester of Minzer and Zheng [42] makes $\text{poly}(\log^d t)$ queries, and a recent refined tester and analysis yield the tight upper bound of $O(\log^d t)$ queries for this parameter regime [36].

[1] As noted in the Encyclopedia of Algorithms [50], the set of homomorphisms from G to H forms an error-correcting code, called the *homomorphism code*. When $G = \mathbb{F}_2^n$ and $H = \mathbb{F}_2$, this corresponds to the well-known Hadamard code. From this perspective, homomorphism testing is equivalent to testing membership in the corresponding code.

[2] We use additive notation for all group operations, even for non-abelian groups (as opposed to using $\cdot$ and $*$), as it enables compact summation notation (specifically, $\sum$ and $\bigoplus$) and leads to more concise and readable expressions.

While both lower bounds [31, 10] apply to the weakest model of online adversary (fixed-rate with erasures), the algorithms of [10, 4, 42, 60] are resilient even in the strongest model considered (budget-managing with corruptions).

These developments raise a natural question: Can one design testers resilient to online manipulations for the other natural generalization of linearity – namely, group homomorphism?[3] We answer this question affirmatively by designing testers that are resilient to the strongest type of online adversary in our framework – budget-managing with corruptions. Our main result is that homomorphism of functions $f : G \to H$ can be tested with $O(1/\varepsilon + \log t)$ queries in the presence of a t-online corruption adversary for all groups G and H. This bound is optimal: even without manipulations, $\Omega(1/\varepsilon)$ queries are required, and the $\Omega(\log t)$ lower bound of [31] holds for the special case $G = \mathbb{F}_2^n$ and $H = \mathbb{F}_2$.

Since the lower bound applies to specific groups G and H, a natural next question is whether some groups admit better testers. We obtain improved group-specific query bounds for many important families of groups. In particular, we design an online-manipulation-resilient homomorphism tester that makes $O(1/\varepsilon + E(G))$ queries, where $E(G)$ is the expected number of samples from G needed to generate G. This tester is optimal, for instance, when G is a cyclic, symmetric, alternating, or simple group. For vector spaces over finite fields of prime order, we obtain nearly tight bounds: when $G = \mathbb{F}_p^n$ and $H = \mathbb{F}_p^r$, homomorphism can be tested with $O(1/\varepsilon + \log_p t)$ queries, and we prove a matching $\Omega(\log_p t)$ lower bound for the case $r = 1$; when $H = \mathbb{F}_q^r$ where $q \neq p$, testing reduces to checking whether f is identically zero.

Our results demonstrate that group homomorphism – a fundamental algebraic property – admits efficient and manipulation-resilient testers across the board: we give a general optimal tester for all finite groups and complement it with sharper bounds for prominent families of domains and codomains.

1.1 Our results

Our main result is an *optimal* online-manipulations-resilient homomorphism tester of $f : G \to H$ for all groups G and H. As all our testers, it works in the strongest online adversary model considered: budget-managing with corruptions.

▶ **Theorem 1.1.** *There exists a constant $c > 0$ such that for all finite groups $(G, +)$ and $(H, \oplus)$, all $\varepsilon \in (0, 1)$ and $t \leq c \cdot \min\left\{\varepsilon^2, 1/\log^2 |G|\right\} \cdot |G|$, there exists an ε-tester for group homomorphism of functions of the form $f : G \to H$ that works in the presence of every budget-managing t-online erasure (or corruption) adversary and makes $O\left(\frac{1}{\varepsilon} + \log_2 t\right)$ queries. For erasures, the tester has 1-sided error.*

Our result recovers the celebrated $O(1/\varepsilon)$ bound by [15, 11] for testing homomorphism in the standard property testing model, albeit with a different tester. Our tester, which we call Random Signs Test, can be viewed as *lifting* the linearity testers developed by [31, 10, 4] for functions $f : \mathbb{F}_2^n \to \mathbb{F}_2$ to work for general groups G and H. As noted by [31], the original [15] tester breaks in the online manipulations model, and as we explain in Section 1.2, the known online-manipulation-resilient testers for the case $G = \mathbb{F}_2^n$ and $H = \mathbb{F}_2$ are not sound for general groups. Let $(G, +)$ be the domain group and $(H, \oplus)$ be the codomain group. For

[3] Importantly, efficient testability of a property $\mathcal{P}$ in the standard property testing model does not guarantee that $\mathcal{P}$ is testable with online adversary. For instance, [31] show that *sortedness* of integer sequences – despite having many efficient testers in the standard model [20, 19, 48, 13, 16, 49, 9] – is not testable even with a single online erasure per query, regardless of the number of queries.

the case without manipulations, our tester picks k points (for any even k) $x_1, \ldots, x_k$ from the domain group G and k signs $\sigma_1, \ldots, \sigma_k$ from $\{+, -\}^k$ uniformly and independently at random. Then it obtains an element $a = \sigma_1 x_1 + \cdots + \sigma_k x_k$, where $\sigma_i x_i$ denotes x_i when $\sigma_i = +$ and the inverse of x_i (in G) when $\sigma_i = -$. Finally, it queries f on $x_1, \ldots, x_k$, and a, and tests whether the homomorphism condition is satisfied for these elements, that is, whether $f(a) = \sigma_1 f(x_1) \oplus \cdots \oplus \sigma_k f(x_k)$, where $\sigma_i = -$ denotes the inverse in the group H. The manipulation-resilient version of the tester first samples a reserve of elements of G and then runs Random Signs Test with carefully selected k and elements $x_1, \ldots, x_k$ from the reserve. We provide a technical overview of the tester and its analysis in Section 1.2. Section 3 presents and analyzes Random Signs Test, and Section 5 – a manipulation-resilient version of Random Signs Test.

1.1.1 Group-specific homomorphism testing

Our first group-specific result, Theorem 1.3, gives a homomorphism tester with complexity expressed in terms of the parameter $E(G)$, defined next.

▶ **Definition 1.2** ([47]). *For all finite groups $(G, +)$, let $E(G)$ be the expected number of independent and uniform samples from G needed to obtain a generating set for G.*

This group-specific tester has an additional advantage of being *sample-based*, i.e., querying only independent and uniform elements of G. Its guarantees are given in Theorem 1.3 and proved in the full version.

▶ **Theorem 1.3** (Group-specific sample-based tester). *There exists a constant $c > 0$ such that for all finite groups $(G, +)$ and $(H, \oplus)$, all $\varepsilon \in (0, 1)$ and $t \leq c \cdot \min\left\{\varepsilon^2, 1/E(G)^2\right\} \cdot |G|$, there exists a sample-based ε-tester for group homomorphism of functions of the form $f : G \to H$ that works in the presence of every t-online erasure (or corruption) budget-managing adversary and makes $O(1/\varepsilon + E(G))$ queries. For erasures, the tester has 1-sided error.*

In the model without manipulations, Goldreich and Ron [25] gave a sample-based homomorphism tester with $O(1/\varepsilon + \log |G|)$ queries and showed this bound is tight for some groups G and H (in particular, $G = \mathbb{F}_2^n, H = \mathbb{F}_2$). The query bound in Theorem 1.3 improves on the bound in [25] for many natural groups and is tight when $E(G)$ is constant – e.g., for cyclic groups of prime order $\mathbb{Z}_p$, symmetric groups S_n and alternating groups A_n [18], and, more generally, finite simple groups [38]. This gives us the following corollary.

▶ **Corollary 1.4** (Homomorphism tester for simple groups). *There exists a constant $c > 0$ such that for all finite groups $(G, +)$ and $(H, \oplus)$, where G is a finite simple group, all $\varepsilon \in (0, 1)$ and all $t \leq c \cdot \varepsilon^2 |G|$, there exists a sample-based ε-tester for group homomorphism of functions of the form $f : G \to H$ that works in the presence of a t-online-erasure adversary and makes $\Theta(1/\varepsilon)$ queries.*

To improve the bound in [25] and prove Theorem 1.3, we analyze the subgroup generated by the sample S, rather than only partial sums that can be obtained from S, obtaining better group-specific bounds on the number of samples needed to properly learn the input homomorphism. Our analysis relies on work in group theory [18, 38, 47, 39, 40, 41] that relates $E(G)$ from Definition 1.2 to the size of the smallest set generating G and other structural parameters of the group.

Bounds for prime fields. Finally, we consider groups of the form $G = \mathbb{F}_p^n$ and $H = \mathbb{F}_q^r$, where p and q are primes and $n, r \in \mathbb{N}$. Our next result shows that if $p = q$, the query complexity is below the general bound of $O(1/\varepsilon + \log_2 t)$ from Theorem 1.1.

▶ **Theorem 1.5.** *There exists a constant $c > 0$ such that for all primes p, all $n, r \in \mathbb{N}$, all $\varepsilon \in (0,1)$ and $t \leq c \cdot \min\left\{\varepsilon^2, 1/n^2\right\} p^n$, there exists an ε-tester for homomorphism of functions of the form $f : \mathbb{F}_p^n \to \mathbb{F}_p^r$ that works in the presence of every t-online erasure (or corruption) budget-managing adversary and makes $O(1/\varepsilon + \log_p t)$ queries. For erasures, the tester has 1-sided error.*

Next we show that, when the range is $\mathbb{F}_p$ ($r = 1$), the bound in Theorem 1.5 is tight.

▶ **Theorem 1.6.** *For all primes p, there exists $n_0 \in \mathbb{N}$ such that for all $n \in \mathbb{N}, n \geq n_0$ and $\varepsilon \in (0, \frac{p-1}{2p}]$, every t-online erasure-resilient ε-tester for homomorphism of functions of the form $f : \mathbb{F}_p^n \to \mathbb{F}_p$ must make $\Omega(\log_p t)$ queries.*

We defer the proofs of Theorems 1.5 and 1.6 to the full version. Finally, we observe that the case when $p \neq q$ is easy.

▶ Remark 1.7. For all finite groups G and H such that the zero map is the only homomorphism from G to H (e.g., when $G = \mathbb{F}_p^n$ and $H = \mathbb{F}_q^r$, where p and q are distinct primes and $n, r \in \mathbb{N}$), testing if a function $f : G \to H$ is a homomorphism is equivalent to testing if f is identically zero and can be done with $\Theta(1/\varepsilon)$ queries, even in the presence of a t-online manipulation budget-managing adversary.

1.1.2 A general framework for resilience

Resilience to online manipulations creates a fundamental tension: a tester must make queries that are hard for the adversary to predict, yet these may not be the most informative ones. This perspective naturally interpolates between two familiar extremes in property testing: query-based testers, which optimize for information useful for the specified computational task but may be predictable, and sample-based testers, which query f on independent uniformly random elements of G and thus maximize unpredictability. Understanding how to balance these competing goals provides a new lens on property testing.

To formalize this perspective, we need a notion of unpredictability that captures this trade-off. A natural candidate is the notion of *fairness* of Goldreich and Ron [25], which constrains the marginal distribution of each query. A tester is *α-fair* if, at every step, no $x \in G$ is queried with probability more than α. They showed that if a property has an α-fair ε-tester that makes q queries with $\alpha = \Theta(|G|^{-1})$ and constant ε then it also has a sample-based tester (in a sense, a "completely fair" tester) with $|G|^{1-\frac{1}{q}}$ queries. This provides an intriguing trade-off between the number of queries and their predictability. However, fairness is ill-suited for resilience to online manipulations, as it does not account for dependencies between queries. For example, the [15] linearity test is completely fair – each query is uniformly random – yet, as observed by [31], it is vulnerable to online manipulations because its third query is fully determined by the first two. This suggests that marginal notions of unpredictability are insufficient.

The key ingredient in our framework for proving resilience is a new notion of *conditional* unpredictability for testers (Definition 4.2). Informally, a tester is (q, α, β)-unpredictable if, with probability at least $1 - \beta$, *each* of its q queries remains hard to predict even after conditioning on the previous queries. Formally, there exist parameters $\{\alpha_i, \beta_i\}_{i \in [q]}$ with $\sum_{i \in [q]} \alpha_i = \alpha$ and $\sum_{i \in [q]} \beta_i = \beta$ such that for each $i \in [q]$, with probability at least $1 - \beta_i$, the

conditional distribution of the ith query Q_i is α_i-flat, i.e., $\max_{x \in G} \Pr[Q_i = x \mid Q_1, \ldots, Q_{i-1}] \leq \alpha_i$. This sequential, conditional structure is reminiscent of Santha–Vazirani sources from the extractor literature [57].

We show that unpredictability directly yields resilience to online manipulations: regardless of the past manipulations, an unpredictable tester is unlikely to ever query a manipulated entry. In particular, a (q, α, β)-unpredictable tester queries a manipulated location with probability at most $\alpha T + \beta$, where T is the total number of manipulations over the entire execution (see Lemma 4.3). To prove our main results, we first establish soundness of a *core test*, modify it to get an *unpredictable core test* which has the same soundness as the core test along with sufficient unpredictability. To get our final testers, we then amplify the success probability of the unpredictable core test by repetition. We address some nuances of amplification in the online manipulation model – where accumulated manipulations can make later repetitions less reliable – and prove an amplification lemma that specifies the required parameters of the unpredictable core test and the number of repetitions.

1.2 Technical overview of the tester for general groups

The starting point for the classical homomorphism testers, as well as for manipulation-resilient linearity testers, is a core test (equivalently, a local characterization of a property). All core tests we discuss have perfect *completeness:* they always accept homomorphisms. In addition, they need to have good *soundness:* high enough rejection probability of inputs that are far from the desired property. For the standard model, good soundness suffices, since repeating the core test amplifies its rejection probability and yields the desired tester. For the online adversarial model, however, soundness alone is insufficient; the final tester must also be unpredictable enough to prevent the adversary from targeting its queries. As in [31], this is achieved by first building a reservoir of uniformly random queried elements, then using a final query to simulate the core test on a random subset of the reservoir, and finally repeating the resulting procedure as needed.

Our main technical contribution is the design and analysis of a new core test. While its full analysis with the reservoir requires additional ideas, we focus here on why a natural extension of the manipulation-resilient linearity test fails in our setting, how our test is constructed, and the key insights behind its correctness.

Before introducing our tester, we discuss why a natural generalization of online manipulation resilient testers from previous work fails to provide soundness. The $(k+1)$-point linearity test of [31] queries the input function f on k independent and uniformly random elements $x_1, \ldots, x_k$ and their sum $x_{k+1} := \sum_{i \in [k]} x_i$ and checks whether $\sum_{i \in [k]} f(x_i)$ equals $f(\sum_{i \in [k]} x_i)$. Let $(G, +)$ and $(H, \oplus)$ be finite groups and $f : G \to H$ be a function. A natural extension of the [31] test to the homomorphism property is to check whether[4]

$$f(x_1) \oplus f(x_2) \oplus \cdots \oplus f(x_k) \stackrel{?}{=} f(x_1 + x_2 + \cdots + x_k). \tag{1}$$

If f is a homomorphism, then (1) holds for all tuples $(x_1, \ldots, x_k)$, but the converse is false: for $G = H = \mathbb{Z}_{k-1}$ (the additive cyclic group over $k-1$ elements) and any homomorphism $h : G \to H$, the shifted homomorphism function $f(x) = h(x) \oplus s$ with shift $s \in H$ also satisfies (1) for all tuples.[5] To resolve this issue, our homomorphism tester uses a more robust local characterization.

---

[4] We use compact notation to represent our sums: (1) can also be written as $\bigoplus_{i \in [k]} f(x_i) = f\left(\sum_{i \in [k]} x_i\right)$. In non-Abelian groups, the order matters. The terms in the sums are ordered according to the index: in increasing index order for $\sum_{i \in [k]} x_i$ or $\bigoplus_{i=1}^{k} y_i$, and in decreasing index order for $\sum_{i=k}^{1} x_i$.

[5] A similar phenomenon occurs for functions $f : \mathbb{F}_2^n \to \mathbb{F}_2$: the k-point tester with even k works for linearity, but odd k produces a test for *affinity* instead.

Next, we discuss our desired soundness guarantee and sketch a simple argument showing that many NO instances are rejected by the $(k+1)$-point test with sufficient probability. (The example above shows that this tester fails on functions that are really far from any homomorphism, but it turns out it works well for functions which are closer to a homomorphism). Let h_f be a closest homomorphism to f and ε_f be the distance from f to h_f. Define $\Delta_f := \{x \in G : f(x) \neq h_f(x)\}$ (so that $|\Delta_f| = \varepsilon_f \cdot |G|$). Since all tuples $(x_1, \ldots, x_k)$ satisfy modified (1) with f replaced by h_f, one could analyze for how many tuples (1) holds for f by comparing f and h_f on each tuple. The following "wishful thinking" argument is an easy assertion when ε_f is small enough, and was used before, e.g., for optimal analysis of low-degree testing over $\mathbb{F}_2$ [14].

Simple case: $\varepsilon_f < \frac{1}{2k}$. We view each point x_i (including the sum x_{k+1}) as a random variable, and ask whether $x_i \in \Delta_f$. Marginally, each x_i is distributed uniformly over G and so this occurs with probability ε_f for all $i \in [k+1]$. Intuitively, the points are "independent enough" so when $\varepsilon_f < \frac{1}{2k}$, w.h.p. at most one of them falls in Δ_f. If exactly one point does, then the tester rejects f (by cancellation rule on the group H). Formally, when $\varepsilon_f < \frac{1}{2k}$ the probability of exactly one x_i falling in Δ_f can be lower bounded by $k\varepsilon_f/2$ using the premise, pairwise independence of the points and inclusion-exclusion (Bonferroni inequality of order 2), as was done, e.g., in [14, 10] to analyze low-degree tests of polynomials over fields. By a union bound, the rejection probability is at most $k\varepsilon_f$ so the analysis above is optimal (up to a factor of 2) for inputs with small distance ε_f.

However, larger values of ε_f lead to more than a single point in Δ_f for a typical tuple, and the guaranteed pairwise independence is no longer enough. Luckily, the $k+1$ points used in (1) are in fact k-wise independent, which is a stronger guarantee for $k > 2$. We demonstrate how to leverage this fact, focusing on the case $H = \mathbb{F}_2$.

The elementary argument (for $\varepsilon_f < \frac{1}{8}$). Let p_m be the probability that m independent and uniformly random elements $z_1, \ldots, z_m \in G$ satisfy $\bigoplus_{i\in[m]} f(z_i) = \bigoplus_{i\in[m]} h_f(z_i)$. Let $\gamma_m = 1 - p_m$ be the probability of an inequality. By definition, $\gamma_1 = \varepsilon_f$. We utilize the k-wise independence by splitting the last point, $x_{k+1} = \sum_{i\in[k]} x_i$, from the rest and comparing the RHS and LHS of (1) with f to those with h_f where the equality holds. In particular, the event that the RHS of f and h_f agree but the LHS are different, leads to rejection of f. Then

$$\begin{aligned}
&\Pr_{x_1,\ldots,x_k\in G}\left[\bigoplus_{i\in[k]} f(x_i) \neq f\Big(\sum_{i\in[k]} x_i\Big)\right] \\
\geq &\Pr_{x_1,\ldots,x_k\in G}\left[\bigoplus_{i\in[k]} f(x_i) \neq \bigoplus_{i\in[k]} h_f(x_i) \ \wedge\ f\Big(\sum_{i\in[k]} x_i\Big) = h_f\Big(\sum_{i\in[k]} x_i\Big)\right] \\
\geq &\Pr\left[\bigoplus_{i\in[k]} f(x_i) \neq \bigoplus_{i\in[k]} h_f(x_i)\right] - \Pr\left[f\Big(\sum_{i\in[k]} x_i\Big) \neq h_f\Big(\sum_{i\in[k]} x_i\Big)\right] = \gamma_k - \gamma_1,
\end{aligned}$$

where the second inequality uses $\Pr[A \wedge B] \geq \Pr[A] - \Pr[\bar{B}]$ for two events A, B (where $\bar{B}$ is the complement of B). The equality is since $(x_1, \ldots, x_k)$ are i.i.d. samples taken uniformly at random from G.

From here on, the goal is to lower bound γ_k. To showcase the argument we restrict ourselves to $H = \mathbb{F}_2$ (which in particular captures linearity over $\mathbb{F}_2$), and discuss the extension later. Here, $f(x) - h_f(x) = 1$ if $x \in \Delta_f$, and otherwise $f(x) = h_f(x)$. That is, γ_k is the probability that there is an odd number of indices $i \in [k]$ for which $x_i \in \Delta_f$. Since these points are independent and each has probability ε_f to fall inside Δ_f, this equals to the

probability of a binomial random variable $Z \sim Bin(k, \varepsilon_f)$ being odd, which is known to equal $\frac{1-(1-2\varepsilon_f)^k}{2}$. This expression has a known lower bound of $\Omega(\min\{k\varepsilon_f, 1\})$, which absorbs the subtraction of $\gamma_1 = \varepsilon_f$ for any $\varepsilon_f \leq 1/8$ and completes the proof.[6]

To generalize this argument, note that $H = \mathbb{F}_2$ is essentially the worst-case scenario, as "coincidental cancellations" are most probable when $f(x) - h_f(x) \in \{0, 1\}$. In Section 3.1 we prove the claim for our modified tester, described shortly, and any codomain H, by analyzing an inductive recurrence directly on p_m (rather than $\gamma_m = 1 - p_m$).

The random signs test. We make the local characterization more robust by adding *signs*. We rely on the inverse operation, which exists in every group, and use $-x$ to express the addition of the inverse of x for both groups, G and H. Since every homomorphism satisfies $f(-x) = -f(x)$ for all $x \in G$, it respects not only summations of elements but signed summations as well. The corresponding tester is this: given a function $f : G \to H$, choose uniformly at random (and independently) a tuple $(x_1, \ldots, x_k, \sigma_1, \ldots, \sigma_k)$ consisting of k elements $x_1, \ldots, x_k \in G$, and k signs $\sigma_1, \ldots, \sigma_k \in \{+, -\}$. Finally, check whether

$$\bigoplus_{i\in[k]} \sigma_i f(x_i) \stackrel{?}{=} f(\sum_{i\in[k]} \sigma_i x_i). \tag{2}$$

As before, if f is a homomorphism, all tuples satisfy equation (2). The difference is that now, when k is even, the converse holds as well.[7] To analyze the soundness of Random Signs Test, we consider two cases. For $\varepsilon_f < 1/8$, the elementary argument (Lemma 3.2) suffices. To prove this tester rejects functions with $\varepsilon_f \geq 1/8$ (unlike the unmodified k-point tester), we show the contrapositive via a corrector argument based on that of [15, 11].

In the corrector method, a corrector function $g(x)$ is defined as the value satisfying the most instances of equation (2), where each instance corresponds to a signed tuple for which $\sum_{i\in[k]} \sigma_i x_i = x$. Differently put, each tuple of elements and signs that sum to x "votes" for a value it wishes $f(x)$ to be – then $g(x)$ is defined as the plurality of these votes. To prove soundness, one shows that if the test rejects f with probability $\mu \leq 1/10$, then two key assertions follow: (a) f is 2μ-close to g; (b) g is a homomorphism. The two assertions are proved in Lemmas 3.12 and 3.13, respectively, with the latter employing the probabilistic method. Both rely on an auxiliary step, Lemma 3.11, which shows that the choice of $g(a)$ as a plurality of votes, is in fact a strong majority of at least $1 - 2\mu$ of the votes, for every input $a \in G$.

We focus here on the proof of this step, which demonstrates the techniques of the full argument, and the role of random signs. The proof analyzes the collision probability – the probability that two signed tuples $(\sigma, x), (\tau, y)$ that sum to a vote for the same value for $f(a)$.

---

[6] We use $\frac{1-(1-2\varepsilon_f)^k}{2} = \Omega(\min\{k\varepsilon_f, 1\})$ with a specific constant, by applying [10, Lemma 3.1].

[7] Indeed, if f always satisfies equation (2), then given $x_1, x_2 \in G$ we use the tuple with elements x_1, x_2 and $x_j = 0$ for $j \geq 3$, and signs $\sigma_1 = \sigma_2 = ' + '$, and σ_j for $j \geq 3$ alternates between addition and subtraction. The $k - 2$ alterations of $+0, -0$ cancel out on the RHS, and similarly the alterations of $+f(0) - f(0)$ cancel out on the LHS (even if $f(0) \neq 0_H$), which asserts that $f(x_1) + f(x_2) = f(x_1 + x_2)$. This holds for any $x_1, x_2 \in G$, showing that f is a homomorphism.

By re-arranging, we get

$$\underbrace{\bigoplus_{i\in[2k]} \sigma_i f(x_i)}_{S_{\sigma,x}} = \underbrace{\bigoplus_{i\in[2k]} \tau_i f(y_i)}_{S_{\tau,y}} \iff$$

$$\underbrace{\bigoplus_{i=0}^{k-1} -\tau_{k-i} f(y_{k-i}) \oplus \bigoplus_{i\in[k]} \sigma_i f(x_i)}_{S_1} = \underbrace{\bigoplus_{i=k+1}^{2k} \tau_i f(y_i) \oplus \bigoplus_{i=0}^{k-1} -\sigma_{2k-i} f(x_{2k-i})}_{S_2}.$$

Crucially, the tuples underlying the sums S_1, S_2 are each a mix from (σ, x) and (τ, y) and we show that both have the distribution of the random sign test, but they sum (in G) to the same element, a'. Therefore $\Pr[S_{\sigma,x} \neq S_{\tau,y}] = \Pr[S_1 \neq S_2] \leq \Pr[S_1 \neq f(a') \vee S_2 \neq f(a')] \leq \Pr[S_1 \neq f(a')] + \Pr[S_2 \neq f(a')] = 2\mu$.

The necessity of random signs is apparent: without them, the sums $S_{\sigma,x}, S_{\tau,y}, S_1, S_2$ correspond to tuples with different sign vector, failing to relate the probability of this event to μ (the rejection probability of the test). By adding random signs, each element has a symmetric role in the test, which leaves the connection between test and equation even after one rearranges the elements, which is a delicate matter especially in non-abelian groups. Similar issues arise for odd k, as there is no way to split the two summations "down the middle". Both technical requirements appear to be tight: we have seen a choice for G, H where fixed signs (say, all are $+$) fail to characterize homomorphisms. Yet for $G = \mathbb{F}_2^n, H = \mathbb{F}_2$ the signs have no meaning ($x = -x$ always), and our test coincides with the simple k-point test, which requires an even value of k to test linearity (otherwise, it tests affinity).

1.3 Additional related work

Injecting more randomness vs. derandomization. A significant effort [59, 55, 29, 12, 58] went into reducing the number of random bits used by homomorphism tests. Interestingly, the online manipulation model presents a contrasting goal: increasing randomness to make future queries less predictable to the adversary. Indeed, Kelman, Linder and Raskhodnikova [34] show that for certain properties, testing in the online model requires exponentially more random bits than in the offline setting (even with erasures), despite comparable query complexity. This increased randomness requirement motivates the design of testers that are more *sample-based*, i.e., use uniformly random labeled samples in place of some arbitrary (potentially adaptive) queries. This trend aligns with learning-theoretic settings, where labeled samples are typically modeled as less costly than arbitrary queries.

Low-soundness regime. Our work focuses on the *high-soundness* regime (which relates to unique decoding), showing that if the input function f is accepted by the tester with high probability, then f is close to a homomorphism. In contrast, in the *low-soundness* regime, the goal is to show that small, yet non-trivial, acceptance probability by the tester implies the input function has a non-trivial structure resembling that of a homomorphism. Homomorphism testing in the low-soundness regime has been extensively studied. Only Khot and Mittal [37] use a k-point tester and their work is specific to linearity over the *biased* hypercube. Their tester chooses points according to Hamming weight rather than uniformly; it is unclear whether it is resilient to online manipulations, since the adversary can manipulate points that are more likely to be queried.

Concurrent work. Concurrently with our work, Mittal and Roy [44] presented a general framework for homomorphism testing in the low-soundness regime and applied it to various choices of domain group G and codomain group H. Specifically, for the case of vector spaces over prime fields, [44] presented a k-point test with random coefficients, identical to the test we design for the special case of such G, H (which appears on the full version). Curiously, the same algorithm is well-suited for both settings (for this specific choice of G and H): the low-soundness regime in the standard model and high soundness in the online manipulations model. The general version of their k-point test chooses a k-tuple from G according to a weight that, roughly, comes from the order of each of its elements in the group G. In contrast, our tester picks k-tuples uniformly at random for any choice of G and H to evade adversarial manipulations.

1.4 Open questions

Our main results are in a sense orthogonal: Theorem 1.1 applies to all finite groups G and H, regardless of their structure, precisely quantifying the dependence on t. In contrast, Theorem 1.3 shows that $E(G)$ samples, rather than $\log|G|$ samples, suffice to generate the entire domain group G and successfully apply the learn-and-test approach. Combining both aspects should yield stronger results in the online manipulation model, as in the case of finite fields (Theorem 1.5). More results of this flavor would be interesting, focusing on the following question: given a group G and $t \in \mathbb{N}$, how many random samples from G are required to generate w.h.p. a subgroup $G' \subseteq G$ of size $|G'|$ that exceeds t?

A different avenue is utilizing the structure of H and its *compatibility* to G, both for upper and lower bounds. As mentioned in Remark 1.7, H could reduce the complexity dramatically when it is incompatible with G. Is it possible to define a measure of compatibility that dictates the complexity of testing homomorphism from G to H? For lower bounds, could we find for each G a group H so that Theorem 1.3 is tight? One candidate would be $H = Aut(G)$, the group of automorphisms from G to G.

Organization. The rest of the paper is organized as follows. We begin in Section 2 with formal definitions of the problem and the online manipulation model. In Section 3, we introduce our core technical contribution, the Random Signs Test (Algorithm 1), and analyze its soundness in the standard (non-adversarial) model. In Section 4, we present the general framework for achieving online manipulation resilience, defining *unpredictable testers* and proving our general amplification lemma (Lemma 4.4). We then apply this framework in Section 5 to prove our main optimal bound for general groups (Theorem 1.1). Finally, our group-specific results, including the bounds for sample-based testers (Theorem 1.3) and for functions over prime fields (Theorems 1.5 and 1.6) are deferred to the full version.

2 Preliminaries

For two groups $(G,+)$ and $(H,\oplus)$, the function $h : G \to H$ is a group *homomorphism* if $h(x_1+x_2) = h(x_1) \oplus h(x_2)$ for all $x_1, x_2 \in G$. In non-Abelian groups, the order matters. The terms in the sums are ordered according to the index: in increasing index order for $\sum_{i\in[k]} x_i$ or $\bigoplus_{i=1}^{k} y_i$, and in decreasing index order for $\sum_{i=k}^{1} x_i$. The set of all group homomorphisms from G to H is denoted $\mathsf{HOM}(G,H)$. The (relative) Hamming *distance* between two functions $f, g : G \to H$ is $dist(f,g) := \Pr_{x\in G}[f(x) \neq g(x)]$. The *distance from* $f : G \to H$ *to a property* (i.e., a set) $\mathcal{P}$ of functions of the same form is $dist(f,\mathcal{P}) := \inf_{g\in\mathcal{P}} dist(f,g)$. For every function $f : G \to H$, let $\varepsilon_f := dist(f, \mathsf{HOM}(G,H))$. A function f is *ε-close to a homomorphism* if $\varepsilon_f \leq \varepsilon$ and is *ε-far from a homomorphism* otherwise.

▶ **Definition 2.1** (Online ε-tester [31, 10]). *Fix $\varepsilon \in (0,1)$. An online ε-tester $\mathcal{T}$ for a property $\mathcal{P}$ that works in the presence of a specified adversary (e.g., t-online erasure budget-managing) is given access to an input function f via oracle access to a related function f' that is initially equal to f and, after each query answered, may be modified by the adversary. For all adversarial strategies of the specified type,*

1. *if $f \in \mathcal{P}$, then $\mathcal{T}$ accepts with probability at least 2/3, and*
2. *if f is ε-far from $\mathcal{P}$, then $\mathcal{T}$ rejects with probability at least 2/3,*

where the probability is over the internal randomness of $\mathcal{T}$. If $\mathcal{T}$ works in the presence of an erasure (resp., corruption) adversary, we refer to it as an online-erasure-resilient (resp., online-corruption-resilient) tester. If $\mathcal{T}$ always accepts every function $f \in \mathcal{P}$, then $\mathcal{T}$ has 1-sided error. *If all queries of $\mathcal{T}$ are chosen uniformly and independently from the domain of f, then $\mathcal{T}$ is* sample-based.

3 Random Signs Test

In this section, we present our Random Signs Test (Algorithm 1) for testing group homomorphism. This test always accepts functions that are homomorphisms because it only rejects if a violation of the homomorphism property is found. We analyze the soundness of the test in Theorem 3.1.

Algorithm 1 Random Signs Test$_k$.

Parameters: $k \in \mathbb{N}$

Input: Query access to a function $f : G \to H$ where $(G, +)$ and $(H, \oplus)$ are finite groups.

1: Draw k independent and uniformly random elements $x_1, x_2, \ldots, x_k$ from G.
2: Draw k independent and uniformly random signs $\sigma_1, \ldots, \sigma_k$ from $\{+, -\}$.
3: Query $f(x_1), f(x_2), \ldots, f(x_k)$ and $f(a)$, where $a \leftarrow \sum_{i\in[k]} \sigma_i x_i$.
▷ $+b = b$ and $-b$ is the inverse of b for both $b \in G$ and $b \in H$.
4: **Reject** if $\bigoplus_{i\in[k]} \sigma_i f(x_i) \neq f(a)$; otherwise, **accept**.

▶ **Theorem 3.1.** *For all $k \in \mathbb{N}$, $\varepsilon \in (0,1)$, finite groups $(G, +)$ and $(H, \oplus)$, and functions $f : G \to H$ such that f is ε-far from being a homomorphism,*

$$\Pr\left[\text{Random Signs Test}_{2k} \text{ rejects}\right] \geq \min\left\{\frac{(2k-3)\cdot\varepsilon}{3}, \frac{1}{16}\right\}. \tag{3}$$

Recall that ε_f denotes the (relative) distance from f to $\mathsf{HOM}(G, H)$. We analyze the soundness of Random Signs Test for small ε_f in Section 3.1 and for large ε_f in Section 3.2. We put these two results together and complete the proof of Theorem 3.1 in Section 3.3.

3.1 Soundness analysis for small ε_f

In this section, we state and prove Lemma 3.2, establishing the soundness of Algorithm 1 for $\varepsilon_f < \frac{1}{8}$.

▶ **Lemma 3.2.** *For all $k \in \mathbb{N}$, finite groups $(G, +)$ and $(H, \oplus)$, and functions $f : G \to H$ with $\varepsilon_f < \frac{1}{8}$,*

$$\Pr\left[\text{Random Signs Test}_{k} \text{ rejects}\right] \geq \min\left\{\frac{(k-3)\cdot\varepsilon_f}{3}, \frac{1}{6}\right\}. \tag{4}$$

To prove Lemma 3.2, we introduce Fixed Test$_k$ (Algorithm 2) which uses a fixed sequence of k signs $\overline{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_k)$, in contrast to the uniform and independent signs from $+, -$ used in Algorithm 1.

Algorithm 2 Fixed Test$_k(\overline{\sigma})$.

Parameters: $k \in \mathbb{N}$, sign sequence $\overline{\sigma} \in \{+,-\}^k$
Input: Query access to a function $f : G \to H$ where $(G,+)$ and $(H,\oplus)$ are finite groups.
1: Draw k independent and uniformly random elements $x_1, x_2, \ldots, x_k$ from G.
2: Query $f(x_1), f(x_2), \ldots, f(x_k)$ and $f(a)$, where $a \leftarrow \sum_{i\in[k]} \sigma_i x_i$.
3: **Reject** if $\bigoplus_{i\in[k]} \sigma_i f(x_i) \neq f(a)$; otherwise **accept**.

Note that Algorithm 2 with $\overline{\sigma}$ set to $+^k$ recovers the tester of [31].

▶ **Lemma 3.3.** *For all $k \in \mathbb{N}$, all sign sequences $\overline{\sigma} \in \{+,-\}^k$, finite groups $(G,+)$ and $(H,\oplus)$, and functions $f : G \to H$ with $\varepsilon_f < \frac{1}{8}$,*

$$\Pr\left[\text{Fixed Test}_k(\overline{\sigma}) \text{ rejects}\right] \geq \min\left\{\frac{(k-3)\cdot\varepsilon_f}{3}, \frac{1}{6}\right\}. \tag{5}$$

We prove Lemma 3.3 after showing that it implies Lemma 3.2.

Proof of Lemma 3.2 assuming Lemma 3.3. Random Signs Test$_k$ (Algorithm 1) chooses the signs $\overline{\sigma}$ independently and uniformly at random from $\{+,-\}^k$. Thus, the rejection probability of Random Signs Test$_k$ is the expected rejection probability of Fixed Test$_k(\overline{\sigma})$ for uniformly random $\overline{\sigma}$. ◀

Proof of Lemma 3.3. Fix a sign sequence $\overline{\sigma}$ in $\{+,-\}^k$. Let $g : G \to H$ be a closest homomorphism to f, i.e., $\Pr_{x\in G}[f(x) \neq g(x)] = \varepsilon_f$. Then Fixed Test$_k(\overline{\sigma})$ always accepts g, since for all $x_1, x_2, \ldots, x_k \in G$,

$$\bigoplus_{i\in[k]} \sigma_i g(x_i) = g\Big(\sum_{i\in[k]} \sigma_i x_i\Big). \tag{6}$$

The core of the analysis is to compare how Algorithm 2 behaves on f and g for each sample. One way Algorithm 2 can reject f is by selecting $x_1, \ldots, x_k$ such that the RHS of (6) remains unchanged when replacing g with f, but the LHS does not.

$$\Pr\left[\text{Fixed Test}_k(\overline{\sigma}) \text{ rejects}\right] \geq \Pr_{x_1,\ldots,x_k\in G}\Big[\bigoplus_{i\in[k]} \sigma_i f(x_i) \neq \bigoplus_{i\in[k]} \sigma_i g(x_i) \bigwedge f\Big(\sum_{i\in[k]} \sigma_i x_i\Big) = g\Big(\sum_{i\in[k]} \sigma_i x_i\Big)\Big]. \tag{7}$$

We consider the probabilities of the complements of the two events – on the LHS and the RHS of (7) – obtaining the complement of the event on the RHS by applying the De Morgan's law:

$$\begin{aligned}
&\Pr[\text{Fixed Test}_k(\overline{\sigma}) \text{ accepts}] \\
&\leq \Pr_{x_1,\ldots,x_k\in G}\Big[\bigoplus_{i\in[k]} \sigma_i f(x_i) = \bigoplus_{i\in[k]} \sigma_i g(x_i) \bigvee f\Big(\sum_{i\in[k]} \sigma_i x_i\Big) \neq g\Big(\sum_{i\in[k]} \sigma_i x_i\Big)\Big] \\
&\leq \Pr\Big[\bigoplus_{i\in[k]} \sigma_i f(x_i) = \bigoplus_{i\in[k]} \sigma_i g(x_i)\Big] + \Pr\Big[f\Big(\sum_{i\in[k]} \sigma_i x_i\Big) \neq g\Big(\sum_{i\in[k]} \sigma_i x_i\Big)\Big],
\end{aligned} \tag{8}$$

where (8) holds by a union bound. The second term of (8) is exactly ε_f, as $\sum_{i\in[k]}\sigma_i x_i$ is a uniformly random element from G. Let p_k denote the first term of (8) where the subscript k indicates the number of elements the probability is taken over.

We next use induction on k to prove the following. Let $k_0 = \lfloor 1/\varepsilon_f \rfloor$, then

$$p_k \le \begin{cases} 1 - k\varepsilon_f/e & \text{for } k \in [k_0]; \\ \max\{p_{k_0}, 1/2 + \varepsilon_f\} & \text{for } k > k_0. \end{cases}$$

▷ Claim 3.4 (Base Case). For all $k \le 1/\varepsilon_f$, we have $p_k \le 1 - k\varepsilon_f/e$.

▷ Claim 3.5 (Inductive Step). For all $k > 1$, we have $p_k \le \max\{p_{k-1}, 1/2 + \varepsilon_f\}$.

The proofs of Claim 3.4 and Claim 3.5 are deferred to the full version.

Note that $k_0 = \lfloor 1/\varepsilon_f \rfloor > 1/\varepsilon_f - 1$. Using Claim 3.4 we have $p_{k_0} \le 1 - k_0\varepsilon_f/e < 1 - (1-\varepsilon_f)/e$. If $\varepsilon_f \le 1/8$, then $p_{k_0} \le 0.7$. Using Claim 3.5 we have for all $k \ge k_0$, $p_k \le \max\{p_{k_0}, 1/2 + \varepsilon_f\} \le 0.7$ for $\varepsilon_f \le 1/8$. We complete the proof by noting that $\Pr[\textsf{Fixed Test}_k(\overline{\sigma}) \text{ accepts}] \le p_k + \varepsilon_f$ from (8). Thus, $\Pr[\textsf{Fixed Test}_k(\overline{\sigma}) \text{ rejects}] \ge (1 - p_k) - \varepsilon_f \ge \min\{k\varepsilon_f/e - \varepsilon_f, 0.3 - \varepsilon_f\} \ge \min\{(k-3)\varepsilon_f/3, 1/6\}$. ◀

3.2 Soundness analysis for large ε_f

In this section, we show that Random Signs Test used with *even* k provides soundness proportional to ε_f. Specifically, we prove the following lemma.

▶ **Lemma 3.6.** *For all $k \in \mathbb{N}$, finite groups $(G, +)$ and $(H, \oplus)$, and functions $f : G \to H$,*

$$\Pr[\textsf{Random Signs Test}_{2k} \textit{ rejects}] \ge \min\left\{\frac{\varepsilon_f}{2}, \frac{1}{10}\right\}. \tag{9}$$

For ease of notation, let us define $\mu := \Pr[\textsf{Random Signs Test}_{2k} \text{ rejects } f]$. Lemma 3.6 immediately follows from the following lemma.

▶ **Lemma 3.7.** *If $\mu < \frac{1}{10}$, then f is 2μ-close to a homomorphism.*

Before proving Lemma 3.7, we explain why it implies Lemma 3.6. By Lemma 3.7, if $\mu < 1/10$ then f is 2μ-close to some homomorphism. Since f has distance ε_f to $\textsf{HOM}(G, H)$, we get $2\mu \ge \varepsilon_f \implies \mu \ge \frac{\varepsilon_f}{2}$. Thus, $\mu \ge \min\left\{\frac{\varepsilon_f}{2}, \frac{1}{10}\right\}$. Next we prove Lemma 3.7.

Proof of Lemma 3.7. In the test, the distribution of the element a, as specified in Algorithm 2, Line 2, is uniform over all elements of G, since for each realization of the signs $\sigma_1, \ldots, \sigma_{2k}$ and $x_1, \ldots, x_{2k-1}$, there is a single choice of x_{2k} leading to each a. Define $\text{FIX}(a)$ to be the set of realizations for σ's and x's leading to a,

$$\text{FIX}(a) := \left\{(\sigma_1, \ldots, \sigma_{2k}, x_1, \ldots, x_{2k}) : \textstyle\sum_{i\in[2k]} \sigma_i x_i = a\right\}.$$

We use $(\overline{\sigma}, \overline{x})$ as a shorthand for $(\sigma_1, \ldots, \sigma_{2k}, x_1, \ldots, x_{2k})$. A random tuple from $\text{FIX}(a)$ is k-wise independent in the following sense.

▷ Claim 3.8. For all $a \in G$, all sets of indices $S \subseteq [2k]$ of size $|S| = k$, let $(\overline{\sigma}, \overline{x})$ be a uniformly random tuple from $\text{FIX}(a)$. Then

1. The vector $(\sigma_i, x_i)_{i\in S}$ is a uniformly distributed vector of k signs and k elements from G.
2. The partial sum $\sum_{i\in S} \sigma_i x_i$ is a uniformly distributed element in G.

Proof.

Part 1. Consider a tuple $(\sigma_i, x_i)_{i\in[2k]}$, not necessarily in $\mathrm{FIX}(a)$, and fix a partial assignment that gives values only to $(\sigma_i, x_i)_{i\in S}$. Now choose an index $j \in [2k] \setminus S$ and let (σ_j, x_j) be chosen uniformly at random. Fill up the remaining indices, (σ_i, x_i) for $i \in [2k] \setminus (S \cup \{j\})$ with arbitrary values. Since (σ_j, x_j) is chosen uniformly at random, the sum $\sum_{i=1}^{j} \sigma_i x_i$ is a uniformly random element in G. Also, for any group G, the function $x \to x + g$ is a bijection from G to G for all $g \in G$ (the same holds true for $g + x, x - g, -g + x$). Thus, the rest of the additions and subtractions (for $i = j + 1, \ldots, 2k$) are all bijections and therefore the entire sum $\sum_{i\in[2k]} \sigma_i x_i$ is also a uniformly random element of G. This means that exactly $1/|G|$ of the tuples with a particular partial assignment to $(\sigma_i, x_i)_{i\in S}$ are in fact in $\mathrm{FIX}(a)$. The same argument applies for any partial assignment of $(\sigma_i, x_i)_{i\in S}$, so each such partial assignment produces the exact same number of tuples in $\mathrm{FIX}(a)$.

Part 2. Consider the set F_z which consists of all assignments to $(\sigma_i, x_i)_{i\in S}$ for which $\sum_{i\in S} \sigma_i x_i = z$. Using similar arguments as in the proof of part 1, each set F_z has the same size, and the same number of ways to complete the assignment to a tuple in $\mathrm{FIX}(a)$. Thus, the number of tuples in $\mathrm{FIX}(a)$ for which $\sum_{i\in S} \sigma_i x_i = z$ is the same for each z. ◁

Consider tuples whose signed sums over either their left or right halves equal a fixed value. Define

$$\mathrm{LEFT}(z) := \left\{(\overline{\sigma}, \overline{x}) : \textstyle\sum_{i\in[k]} \sigma_i x_i = z\right\}, \quad \mathrm{RIGHT}(z) := \left\{(\overline{\sigma}, \overline{x}) : \textstyle\sum_{i\in[k]} \sigma_{k+i} x_{k+i} = z\right\}.$$

▶ **Corollary 3.9.** *Consider the sets $S_L = [k]$ and $S_R = [2k] \setminus [k]$. The following process produces a uniformly random tuple in FIX(a):*

1. *Draw a uniformly random element $z \in G$.*
2. *Draw a uniformly random tuple $(\overline{\sigma}^L, \overline{x}^L)$ from LEFT$(a - z)$ and take its "left" part, $(\overline{\sigma}^L, \overline{x}^L)_{i\in S_L}$*
3. *Draw a uniformly random tuple $(\overline{\sigma}^R, \overline{x}^R)$ from RIGHT(z) and take its "right" part, $(\overline{\sigma}^R, \overline{x}^R)_{i\in S_R}$*
4. *Return the tuple $(\overline{\sigma}, \overline{x})$ where $(\overline{\sigma}, \overline{x})_{i\in S_L} = (\overline{\sigma}^L, \overline{x}^L)_{i\in S_L}$ and $(\overline{\sigma}, \overline{x})_{i\in S_R} = (\overline{\sigma}^R, \overline{x}^R)_{i\in S_R}$*

Proof. By the second part of Claim 3.8, the number of tuples $(\sigma_1, \ldots, \sigma_{2k}, x_1, \ldots, x_{2k}) \in \mathrm{FIX}(a)$ such that their right part is in $\mathrm{RIGHT}(z)$ (that is, $\sum_{i\in[k]} \sigma_{k+i} x_{k+i} = z$) is the same for each z. The same holds for the $\mathrm{LEFT}(a - z)$. Since each tuple in $\mathrm{FIX}(a)$ has a unique such pair $(a - z, z)$, the process gives an equal probability for each tuple in $\mathrm{FIX}(a)$, concluding the proof. ◀

▶ **Definition 3.10** (Vote of $(\overline{\sigma}, \overline{x})$ for the value of a)**.** *For all $a \in G$ and $(\overline{\sigma}, \overline{x}) \in$ FIX(a), define $\bigoplus_{i\in[2k]} \sigma_i f(x_i)$ as the* vote of $(\overline{\sigma}, \overline{x})$ for the value of a.

Corrector. Define g to be the local corrector for f, that is, the value that has the most votes as defined in Definition 3.10:

$$g(a) := \mathrm{argmax}_{h\in H} \left\{\Pr_{(\overline{\sigma},\overline{x})\in \mathrm{FIX}(a)} \left[\textstyle\bigoplus_{i\in[2k]} \sigma_i f(x_i) = h\right]\right\} \text{ for all } a \in G.$$

Define the following probabilities, all relating to how close f is to being homomorphic:

$$\eta_a := \Pr_{(\overline{\sigma},\overline{x})\in \mathrm{FIX}(a)}\left[g(a) \neq \bigoplus_{i\in[2k]} \sigma_i f(x_i)\right] \quad \eta := max_a \eta_a \quad \delta := \Pr_{x\in G}\left[f(x) \neq g(x)\right]$$

We follow the standard self-correction paradigm introduced by [15, 11], by showing that if the test fails with a small probability (e.g., $\mu < 1/10$), then: (1) η is bounded; (2) $\delta \leq 2\mu$; and (3) g is a homomorphism. The last part (3) is evidently the harder one, although similar simpler arguments are used to prove (1). Altogether, they show that if the test fails with a small probability, then f is close to being a homomorphism (and in particular, the homomorphism g). By contrapositive, this proves a soundness guarantee for the tester.

We start by formally stating and proving the first assertion.

▶ **Lemma 3.11.** $\eta \leq 2\mu$.

Proof. It suffices to show that $\eta_a \leq 2\mu$ for each $a \in G$. Fix some a and consider two tuples $(\overline{\sigma},\overline{x}), (\overline{\tau},\overline{y})$ chosen independently and uniformly at random from $\mathrm{FIX}(a)$. We show that the probability of the event that the votes of $(\overline{\sigma},\overline{x})$ and $(\overline{\tau},\overline{y})$ for the value of a are equal is close to 1, which implies the most common vote among all tuples in $\mathrm{FIX}(a)$ occurs with probability close to 1. To check whether they give the same vote we manipulate both summations as follows: apply from the left side $-\tau_k f(y_k)\cdots-\tau_1 f(y_1)$ to cancel out the first k summands from the tuple $(\overline{\tau},\overline{y})$, and similarly from the right side for the last k summands of the other tuple. We get the following equivalence:

$$\underbrace{\bigoplus_{i\in[2k]} \sigma_i f(x_i)}_{S_{\sigma,x}} = \underbrace{\bigoplus_{i\in[2k]} \tau_i f(y_i)}_{S_{\tau,y}} \iff$$

$$\underbrace{\bigoplus_{i=0}^{k-1} -\tau_{k-i} f(y_{k-i}) \oplus \bigoplus_{i\in[k]} \sigma_i f(x_i)}_{S_1} = \underbrace{\bigoplus_{i=k+1}^{2k} \tau_i f(y_i) \oplus \bigoplus_{i=0}^{k-1} -\sigma_{2k-i} f(x_{2k-i})}_{S_2}.$$

Importantly, each of the new sums S_1, S_2 contains exactly k summands respectively from $(\overline{\sigma},\overline{x})$ and $(\overline{\tau},\overline{y})$, but with different signs. By Claim 3.8 and independence between $(\overline{\sigma},\overline{x})$ and $(\overline{\tau},\overline{y})$, the tuple of $2k$ signs and $2k$ elements used in S_1 is uniformly random (and similarly for S_2). Since the original tuples are taken from $\mathrm{FIX}(a)$, $\sum_{i\in[2k]} \sigma_i x_i = a = \sum_{i\in[2k]} \tau_i y_i$. By the same reasoning as before but with x_i, y_i instead of $f(x_i), f(y_i)$

$$\textstyle\sum_{i=0}^{k-1} -\tau_{k-i} y_{k-i} + \sum_{i\in[k]} \sigma_i x_i = \sum_{i=k+1}^{2k} \tau_i y_i + \sum_{i=0}^{k-1} -\sigma_{2k-i} x_{2k-i}.$$

Thus, the two tuples defining the sums or votes S_1, S_2 are both in $\mathrm{FIX}(a')$, for the same uniformly random element $a' \in G$. The event $\{S_1 \neq S_2\}$ implies that at least one of the votes did not agree with $f(a')$.

$$\Pr\left[S_1 \neq S_2\right] \leq \Pr\left[S_1 \neq f(a') \vee S_2 \neq f(a')\right] \leq \Pr\left[S_1 \neq f(a')\right] + \Pr\left[S_2 \neq f(a')\right] = 2\mu,$$

where the second inequality is by a union bound, and the last equality holds since both the probabilities are equal to the probability that the check in Line 4 of Algorithm 1 accepts, thus simulating our test. Indeed, $S_i \in \mathrm{FIX}(a')$ for $i = 1, 2$, where a' is a uniformly random element of G. We denote by p_h the probability that a correction using a random tuple $(\overline{\sigma},\overline{x}) \in \mathrm{FIX}(a)$ produces $h \in H$, then:

$$2\mu \geq \Pr\left[S_1 \neq S_2\right] = \Pr\left[S_{\sigma,x} \neq S_{\tau,y}\right] = \sum_{h\in H} p_h(1-p_h) = \sum_{h\in H} (p_h - p_h^2) = 1 - \sum_{h\in H} p_h^2.$$

By definition, $g(a) = \arg\max_{h \in H} p_h$ is the most common vote or correction outcome (ties broken arbitrarily). We have $1 - \eta_a = p_{g(a)} = p_{g(a)} \cdot \sum_{h \in H} p_h = \sum_{h \in H} p_h p_{g(a)} \geq \sum_{h \in H} p_h^2 \geq 1 - 2\mu$, which concludes the proof. ◀

The next simple lemma shows the second easy assertion as a corollary of the first.

▶ **Lemma 3.12.** *If $\mu < 1/10$, then $\delta \leq 2\mu$.*

Proof. For each element $a \in G$ we have high agreement for corrections, and in particular $\eta_a \leq 1/2$. Denote the set where f and g disagree by $\Delta := \{x \in G : f(x) \neq g(x)\}$, noting that $\delta = |\Delta| / |G|$. Then

$$\mu = \Pr_{\substack{a \in G \\ (\sigma,x) \in \mathrm{FIX}(a)}} \left[f(a) \neq \bigoplus_{i \in [2k]} \sigma_i f(x_i) \right] = \frac{1}{|G|} \sum_{a \in G} \Pr_{(\sigma,x) \in \mathrm{FIX}(a)} \left[f(a) \neq \bigoplus_{i \in [2k]} \sigma_i f(x_i) \right].$$

The latter expression is only reduced if we take a subset of (positive) summands, only those from $\Delta \subseteq G$. It further reduces if we replace each event of the sum differing from $f(a)$ with the subevent that it specifically equals $g(a)$ (this is indeed a subevent for any $a \in \Delta$). We get

$$\mu \geq \frac{1}{|G|} \sum_{a \in \Delta} \Pr_{(\sigma,x) \in \mathrm{FIX}(a)} \left[g(a) = \bigoplus_{i \in [2k]} \sigma_i f(x_i) \right] = \frac{1}{|G|} \sum_{a \in \Delta} (1 - \eta_a) \geq \frac{1}{|G|} \cdot |\Delta| \cdot \frac{1}{2} = \frac{\delta}{2},$$

where the last inequality uses $1 - \eta_a \geq 1/2$ for any $a \in \Delta$. This concludes the proof. ◀

The last assertion uses the probabilistic method and the accuracy of corrections shown above, to prove that g is a homomorphism.

▶ **Lemma 3.13.** *If $\mu < 1/10$, then g is a homomorphism.*

Proof. Given $a, b \in G$, we need to show that $g(a) \oplus g(b) = g(a + b)$. Consider the following process:

1. Draw a uniformly random $r \in G$.
2. Draw a uniformly random tuple $(\overline{\sigma}, \overline{x}) \in \mathrm{LEFT}(a - r) \bigcap \mathrm{RIGHT}(r)$.
3. Let CL be the set "cancelling left" whose left side cancels exactly the right side of the previous tuple, defined as, $CL := \{(\overline{\tau}, \overline{y}) : \forall i \in [k].(\tau_i, y_i) = (-\sigma_{2k+1-i}, x_{2k+1-i})\}$, and choose uniformly at random a tuple $(\overline{\tau}, \overline{y})$ from $CL \bigcap \mathrm{RIGHT}(r + b)$.

Applying Corollary 3.9 three times, note that $(\overline{\sigma}, \overline{x})$ is a uniformly random tuple in $\mathrm{FIX}(a)$, the tuple $(\overline{\tau}, \overline{y})$ is a uniformly random tuple in $\mathrm{FIX}(b)$. Furthermore, we define the tuple $(\overline{\phi}, \overline{z})$ that has the left side taken from $(\overline{\sigma}, \overline{x})$ and the right side taken from $(\overline{\tau}, \overline{y})$, that is $(\phi_i, z_i) = (\sigma_i, x_i)$ for $i \in [k]$ and $(\phi_i, z_i) = (\tau_i, y_i)$ for $i \in [2k] \setminus [k]$. Applying Corollary 3.9 again, we get that $(\overline{\phi}, \overline{z})$ is a uniformly random tuple from $\mathrm{FIX}(a + b)$. Hence, each of the following three equalities holds with probability at least $1 - \eta$:

$$g(a) = \bigoplus_{i \in [2k]} \sigma_i f(x_i) \qquad g(b) = \bigoplus_{i \in [2k]} \tau_i f(y_i) \qquad g(a+b) = \bigoplus_{i \in [2k]} \phi_i f(z_i)$$

Next, recall we specifically chose the left side of $(\overline{\tau}, \overline{y})$ to cancel out the right side of $(\overline{\sigma}, \overline{x})$. This cancellation also happens when applying the function f, since pairs of summands cancel each other from inside out. The following equality thus holds with probability 1:

$$\bigoplus_{i=k+1}^{2k} \sigma_i f(x_i) \oplus \bigoplus_{i \in [k]} \tau_i f(y_i) = \sigma_{k+1} f(x_{k+1}) \oplus \cdots \oplus \sigma_{2k} f(x_{2k}) \oplus \tau_1 f(y_1) \oplus \cdots \oplus \tau_k f(y_k) = e_H.$$

Over the random choice of tuples, the event that all 4 equalities above hold has probability at least $1-3\eta \geq 1-6\mu > 0$ (by applying Lemma 3.11, a union bound, and the premise $\mu < 1/6$). This event implies that $g(a) \oplus g(b) = g(a+b)$, which also occurs with positive probability. However, the last equality does not depend on the probability space, and therefore it must hold with probability 1. ◀

This completes the proof of Lemma 3.7. ◀

3.3 Completing the soundness analysis

Proof of Theorem 3.1. Since f is ε-far from being a homomorphism, $\varepsilon_f \geq \varepsilon$. When $\varepsilon_f \geq 1/8$, then by Lemma 3.6, we get $\Pr[\textsf{Random Signs Test}_{2k} \text{ rejects}] \geq \min\left\{\frac{\varepsilon_f}{2}, \frac{1}{10}\right\} \geq \frac{1}{16}$. On the other hand, if $\varepsilon_f < 1/8$, then by Lemma 3.2, $\Pr[\textsf{Random Signs Test}_{2k} \text{ rejects}] \geq \min\left\{\frac{(2k-3)\cdot\varepsilon_f}{3}, \frac{1}{6}\right\} \geq \min\left\{\frac{(2k-3)\cdot\varepsilon}{3}, \frac{1}{6}\right\}$. Combining the two cases, we get

$$\Pr[\textsf{Random Signs Test}_{2k} \text{ rejects}] \geq \min\left\{\frac{(2k-3)\cdot\varepsilon}{3}, \frac{1}{6}, \frac{1}{16}\right\} \geq \min\left\{\frac{(2k-3)\cdot\varepsilon}{3}, \frac{1}{16}\right\}. \quad ◀$$

4 General framework for online manipulation resilience

The previous section provides a sound *core test* for homomorphism. To make this core test resilient to online manipulations, we now introduce the general framework used to build our final testers. This framework, based on the concept of *unpredictability* and an amplification lemma, provides a general recipe for converting a core test with certain unpredictability properties, i.e., an *unpredictable core test*, to one robust to online manipulations. In subsequent sections, we show how to modify our core test into an unpredictable core test and then apply this framework to prove our main results.

4.1 Unpredictable test

We define flatness of a distribution, which can also be seen as a bound on its "min-entropy".[8]

▶ **Definition 4.1** (α-flat). *A distribution ν on domain D is α-flat if* $\max_{z\in D} \nu(z) \leq \alpha$.

The following definition, loosely speaking, requires that other than with probability β of "losing its unpredictability", a q-query test maintains "aggregated flatness" of α. Later in Lemma 4.4, we show that such a tester is resilient to online manipulations.

▶ **Definition 4.2** (Unpredictable tester). *A tester making q sequential queries $x_1, \ldots, x_q$ to a function $f : D \to R$ is* ***(q, α, β)-unpredictable*** *(in the standard model, without an adversary) if there exist nonnegative flatness parameters $(\alpha_1, \ldots, \alpha_q)$ and failure probabilities $(\beta_1, \ldots, \beta_q)$ satisfying the following two conditions.*

1. ***Sequential unpredictability:*** *For every input function $f : D \to R$, the following holds. For each $j \in [q]$, let ν_j be the probability distribution of the j-th query x_j conditioned on all the previous queries $(x_1, \ldots, x_{j-1})$. Then, for each $j \in [q]$, with probability at least $1-\beta_j$ (taken over the randomness of the previous $j-1$ queries), ν_j is α_j-flat:*

$$\Pr_{x_1,\ldots,x_{j-1}}[\nu_j \text{ is } \alpha_j\text{-flat}] \geq 1-\beta_j.$$

2. ***Aggregate parameters:*** *The sums of flatness parameters and of the failure probabilities are bounded, satisfying $\sum_{j\in[q]} \alpha_j \leq \alpha$ and $\sum_{j\in[q]} \beta_j \leq \beta$.*

[8] A distribution is α-flat if and only if its min-entropy is at least $\log(1/\alpha)$.

▶ **Lemma 4.3** (Encountering a manipulation). *Let $\mathcal{A}$ be an algorithm that runs r independent iterations of a (q, α, β)-unpredictable test. Fix an iteration $i \in [r]$, and let E_i be an event completely determined by the first $i-1$ iterations. Then, for all input functions f and t-online budget-managing adversarial strategies, the probability that a manipulation is encountered in iteration i, conditioned on the event E_i, is at most $\alpha qrt+\beta$. (Note that qrt is the manipulation budget for the entire algorithm.)*

Proof. Fix an adversarial strategy. Since $\mathcal{A}$ runs a (q, α, β)-unpredictable test, there exist flatness parameters $(\alpha_1, \ldots, \alpha_q)$ and error terms $(\beta_1, \ldots, \beta_q)$ satisfying the conditions in Definition 4.2. Let M_i be the event that a manipulation was encountered in iteration i. For all $j \in [q]$, let M_{ij} be the event that the j-th query of iteration i is to a manipulated point.

We now bound the probability $\Pr[M_{ij}|E_i]$ for a single i. To do this, consider the sample space of all execution histories of $\mathcal{A}$ up to iteration $i-1$. The event E_i is simply a collection of such histories. Fix a specific execution history h, which fixes all queries made by $\mathcal{A}$ and all manipulations made by the adversary in the previous $i-1$ iterations.

Let G_j be the good event that the distribution ν_j is α_j-flat. Then $\Pr[\overline{G_j}|h] \le \beta_j$. Suppose the event G_j occurred, and let $\mathcal{M}_j$ be the set of manipulated entries when query j is made. Then $|\mathcal{M}_j| \le qrt$. Thus, the probability of the event M_{ij} conditioned on G_j and h,

$$\Pr[M_{ij} \mid G_j, h] = \sum_{z \in \mathcal{M}_j} \Pr_{x_j \sim \nu_j}[x_j = z] \le \sum_{z \in \mathcal{M}_j} \alpha_j \le \alpha_j qrt.$$

The probability of the event M_{ij} conditioned on the history h is bounded by

$$\begin{aligned}\Pr[M_{ij} \mid h] &= \Pr[M_{ij} \mid G_j, h]\Pr[G_j \mid h] + \Pr[M_{ij} \mid \overline{G_j}, h]\Pr[\overline{G_j} \mid h] \\ &\le \Pr[M_{ij} \mid G_j, h] + \Pr[\overline{G_j} \mid h] \le \alpha_j qrt + \beta_j.\end{aligned}$$

By a union bound over the events M_{ij} for all $j \in [q]$, the probability of the event M_i conditioned on h is

$$\Pr[M_i \mid h] \le \sum_{j \in [q]} \Pr[M_{ij}|h] = \sum_{j \in [q]} \left(\alpha_j qrt + \beta_j\right) \le \alpha qrt + \beta.$$

Now, consider all execution histories that satisfy the event E_i.

$$\begin{aligned}\Pr[M_i \mid E_i]\Pr[E_i] = \Pr[M_i \cap E_i] &= \sum_{h \in E_i} \Pr[M_i \cap h] = \sum_{h \in E_i} \Pr[M_i|h]\Pr[h] \\ &\le (\alpha qrt + \beta) \cdot \sum_{h \in E_i} \Pr[h]\end{aligned}$$

$$\implies \Pr[M_i \mid E_i]\Pr[E_i] \le (\alpha qrt + \beta) \cdot \Pr[E_i] \implies \Pr[M_i \mid E_i] \le \alpha qrt + \beta. \qquad \blacktriangleleft$$

4.2 General amplification lemma

In this section, we prove a general amplification lemma. This result provides a recipe for converting an unpredictable core test into an online manipulation-resilient tester. Specifically, we show that if a (q, α, β)-unpredictable test satisfies a condition linking its soundness $p_{\mathsf{rej}}(\varepsilon)$ (probability of rejecting $f \notin \mathcal{P}$ as a function of ε) to its aggregate unpredictability parameters (α, β) and the adversary's manipulation rate t, then running the test $O(1/p_{\mathsf{rej}}(\varepsilon))$ times yields a new tester with a constant soundness guarantee against the adversary.

▶ **Lemma 4.4** (General Amplification for Online Resilience). *Let $\mathcal{P}$ be a property (a set of functions $f : D \to R$). Let $\mathcal{T}$ be a (q, α, β)-unpredictable, 1-sided error ε-test for property $\mathcal{P}$. Suppose $\mathcal{T}$ rejects all functions that are ε-far from $\mathcal{P}$ with probability at least $p_{\mathsf{rej}}(\varepsilon)$. Let $\mathcal{A}$ be the algorithm that runs $\lceil 3/p_{\mathsf{rej}}(\varepsilon)\rceil$ independent iterations of test $\mathcal{T}$ and accepts iff each iteration accepts or (in the case when manipulations are erasures) encounters an erasure. Algorithm $\mathcal{A}$ is an ε-tester for property $\mathcal{P}$ that works in the presence of every t-online manipulation budget-managing adversary if*

$$\alpha q t \cdot \left\lceil \frac{3}{p_{\mathsf{rej}}(\varepsilon)} \right\rceil + \beta \leq \frac{p_{\mathsf{rej}}(\varepsilon)}{12}. \tag{10}$$

Moreover, if the manipulations are erasures, then $\mathcal{A}$ has 1-sided error.

Proof. Let $r = \lceil 3/p_{\mathsf{rej}}(\varepsilon)\rceil$ be the number of iterations of the test $\mathcal{T}$ run by $\mathcal{A}$.

Completeness. Consider $f \in \mathcal{P}$. The tester $\mathcal{A}$ fails on f only if one of the iterations rejects. This can only happen because of a manipulation, since $\mathcal{T}$ has 1-sided error (it always accepts $f \in \mathcal{P}$ when there are no manipulations). For all $i \in [r]$, let E_i be the trivial event that includes the entire sample space. By Lemma 4.3 instantiated with this E_i and by (10), the probability $\mathcal{A}$ encounters a manipulation in any single iteration is at most

$$\alpha q r t + \beta = \alpha q t \left\lceil \frac{3}{p_{\mathsf{rej}}(\varepsilon)} \right\rceil + \beta \leq \frac{p_{\mathsf{rej}}(\varepsilon)}{12}.$$

By a union bound over all $r = \lceil \frac{3}{p_{\mathsf{rej}}(\varepsilon)} \rceil$ iterations, $\mathcal{A}$ encounters a manipulation with probability at most

$$\frac{p_{\mathsf{rej}}(\varepsilon)}{12} \cdot \left\lceil \frac{3}{p_{\mathsf{rej}}(\varepsilon)} \right\rceil \leq \frac{1}{3}.$$

Therefore, $\mathcal{A}$ errs with probability at most $\frac{1}{3}$. Moreover, when manipulations are erasures, $\mathcal{A}$ always accepts f because it is designed to accept in any iteration where it sees an erasure.

Soundness. Assume f is ε-far from $\mathcal{P}$. Let F be the event that the amplified tester $\mathcal{A}$ fails to reject f. For all iterations $i \in [r]$, let F_i be the event that $\mathcal{A}$ fails on iteration i. Then $F = \bigcap_{i\in[r]} F_i$. Therefore,

$$\Pr[F] = \prod_{i\in[r]} \Pr[F_i \mid F_1 \cap \cdots \cap F_{i-1}]. \tag{11}$$

Next we bound each term in the product in (11). For all $i \in [r]$, let A_i be the event that $\mathcal{T}$ accepts in iteration i, and M_i be the event that a manipulated entry is queried in iteration i. Fix an iteration $i \in [r]$. Then $F_i \subseteq A_i \cup M_i = (A_i \cap \overline{M_i}) \cup M_i$. By a union bound,

$$\Pr[F_i \mid F_1 \cap \cdots \cap F_{i-1}] \leq \Pr[(A_i \cap \overline{M_i}) \mid F_1 \cap \cdots \cap F_{i-1}] + \Pr[M_i \mid F_1 \cap \cdots \cap F_{i-1}]. \tag{12}$$

The first term on the RHS of (12) corresponds to the event when $\mathcal{A}$ encounters no manipulations in iteration i and the unpredictable core test $\mathcal{T}$ accepts. In this case, the view of the algorithm $\mathcal{A}$ is the same as if the function f was not manipulated and conditioning on any event determined by the initial $i-1$ iterations such as $F_1 \cap \cdots \cap F_{i-1}$ doesn't affect the probability of the event $A_i \cap \overline{M_i}$. Since the core test $\mathcal{T}$ rejects f with probability at least $p_{\mathsf{rej}}(\varepsilon)$, this implies $\Pr[(A_i \cap \overline{M_i}) \mid F_1 \cap \cdots \cap F_{i-1}] \leq 1 - p_{\mathsf{rej}}(\varepsilon)$. To bound the second term on the RHS of (12), we use Lemma 4.3 with $E_i = \bigcap_{j\in[i-1]} F_j$ and then (10):

$$\Pr[M_i \mid F_1 \cap \cdots \cap F_{i-1}] \leq \alpha q r t + \beta = \alpha q t \left\lceil \frac{3}{p_{\mathsf{rej}}(\varepsilon)} \right\rceil + \beta \leq \frac{p_{\mathsf{rej}}(\varepsilon)}{12}.$$

By (12), $\Pr[F_i \mid F_1 \cap \cdots \cap F_{i-1}] \le (1 - p_{\mathsf{rej}}(\varepsilon)) + \frac{p_{\mathsf{rej}}(\varepsilon)}{12} = 1 - \frac{11 p_{\mathsf{rej}}(\varepsilon)}{12}$. The above bound holds for all iterations $i \in [r]$. Thus, by (11),

$$\Pr[F] \le \prod_{i \in [r]} \left(1 - \frac{11 p_{\mathsf{rej}}(\varepsilon)}{12}\right) = \left(1 - \frac{11 p_{\mathsf{rej}}(\varepsilon)}{12}\right)^r.$$

Substituting $r = \lceil 3/p_{\mathsf{rej}}(\varepsilon) \rceil$, we know $r \ge 3/p_{\mathsf{rej}}(\varepsilon)$, thus

$$\Pr[F] \le \left(1 - \frac{11 p_{\mathsf{rej}}(\varepsilon)}{12}\right)^{3/p_{\mathsf{rej}}(\varepsilon)} \le e^{\frac{-11 p_{\mathsf{rej}}(\varepsilon)}{12} \cdot \frac{3}{p_{\mathsf{rej}}(\varepsilon)}} = \left(\frac{1}{e}\right)^{11/4} \le \frac{1}{3},$$

where the second inequality holds since $1 - x \le e^{-x}$ for all x. Thus, the failure probability of $\mathcal{A}$ is at most $1/3$, and $\mathcal{A}$ rejects f with probability at least $2/3$. ◀

5 Resilience to online manipulations for general groups

In this section, we prove Theorem 1.1 by lifting the linearity tester in [4], which is based on the manipulation-resilient tester of [10], and the sample-based tester of Goldreich and Ron [25]. Define a parameter $m := 4\lceil \log_2 t + 15/\varepsilon \rceil + 12$. Depending on the value of m, we invoke one of the two different algorithms against the t-online-erasure adversary. Following the case-based strategy of [4], when $2^m \le |G|^{1/4}$ we use Algorithm 4, a lift of the linearity tester of [10] to the general group setting. On the other hand, when $2^m > |G|^{1/4}$, we instead use the sample-based tester by [25]. We analyze the two cases separately. Our main contribution here is in the analysis of the distribution of the element y in Algorithm 3, where we need to handle random signs along with random elements for any group G, whereas previous work did not have to worry about the random signs.

Case 1: $2^m \le |G|^{1/4}$. In this case, we use Algorithm 4 which internally runs Algorithm 3 a constant number of times.

Algorithm 3 Unpredictable Random Signs Test.

Parameters: even $m \in \mathbb{N}$
Input: Query access to $f : G \to H$ where $(G, +)$ and $(H, \oplus)$ are finite groups.
1: Draw m independent and uniformly random elements $x_1, x_2, \ldots, x_m$ from G.
2: Query $f(x_1), f(x_2), \ldots, f(x_m)$.
3: Draw independent and uniformly random signs σ_j for all $j \in [m]$ from the set $\{+, -\}$.
4: Let S be a uniformly random subset of $[m]$ of size $m/2$.
5: Query $f(y)$ where $y \leftarrow \sum_{j \in S} \sigma_j x_j$
6: **Reject** if $\bigoplus_{j \in S} \sigma_j f(x_j) \ne f(y)$. ▷ This implies no erasures were encountered.
7: **Accept**.

Note that Algorithm 3 has the same completeness and soundness as $\mathsf{Random\ Signs\ Test}_{m/2}$. Moreover, Algorithm 3 is also unpredictable as stated in Lemma 5.1 whose proof we defer to the full version.

▶ **Lemma 5.1.** *Fix a finite group G. If $2^m \le |G|^{1/4}$, then Algorithm 3 with parameter m is*

$$\left(m + 1, \frac{m}{2^m} + \frac{1}{\binom{m}{m/2}}, \frac{1}{2^m}\right)\text{-unpredictable.}$$

Algorithm 4 Online Erasure-Resilient Random Signs Test.

Parameters: $\varepsilon \in (0,1)$, $t \in \mathbb{N}$.
Input: Query access to $f : G \to H$ via t-erasure oracle, where $(G,+)$ and $(H,\oplus)$ are finite groups.
1: Set $m = 4\lceil \log_2 t + 15/\varepsilon \rceil + 12$
2: **for** $i \in [48]$ **do**
3: Run Algorithm 3 with parameter m, independent randomness and query access to f.
4: **Reject** if Algorithm 3 rejects.
5: **Accept**.

Proof of Theorem 1.1 when $2^m \leq |G|^{1/4}$. Finally, we prove that Algorithm 4 satisfies Theorem 1.1 by an application of the general amplification for online resilience lemma (Lemma 4.4). The details are deferred to the full version. ◀

Case 2: $2^m > |G|^{1/4}$. In this case, we use the sample-based tester by Goldreich and Ron [25] and use a theorem by Arora, Kelman and Meir [4] about online-manipulation-resilience of sample-based testers to handle this case. The details are deferred to the full version.

References

1 Noga Alon, Tali Kaufman, Michael Krivelevich, Simon Litsyn, and Dana Ron. Testing Reed-Muller codes. *IEEE Transactions on Information Theory*, 51(11):4032–4039, 2005. doi:10.1109/TIT.2005.856958.

2 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998. doi:10.1145/278298.278306.

3 Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. *Combinatorica*, 23(3):365–426, 2003. doi:10.1007/s00493-003-0025-0.

4 Vipul Arora, Esty Kelman, and Uri Meir. On optimal testing of linearity. In Ioana Oriana Bercea and Rasmus Pagh, editors, *2025 Symposium on Simplicity in Algorithms, SOSA 2025, New Orleans, LA, USA, January 13-15, 2025*, pages 65–76. SIAM, 2025. doi:10.1137/1.9781611978315.5.

5 László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *STOC*, pages 21–31, 1991. doi:10.1145/103418.103428.

6 László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991. doi:10.1007/BF01200056.

7 Oren Becker and Michael Chapman. Stability of approximate group actions: uniform and probabilistic. *Journal of the European Mathematical Society*, 25(9):3599–3632, 2022.

8 Mihir Bellare, Don Coppersmith, Johan Håstad, Marcos A. Kiwi, and Madhu Sudan. Linearity testing in characteristic two. *IEEE Transactions on Information Theory*, 42(6):1781–1795, 1996. doi:10.1109/18.556674.

9 Aleksandrs Belovs. Adaptive lower bound for testing monotonicity on the line. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 31:1–31:10, 2018. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.31.

10 Omri Ben-Eliezer, Esty Kelman, Uri Meir, and Sofya Raskhodnikova. Property testing with online adversaries. *ACM Transactions on Computation Theory*, 18(1):1–40, 2026. doi:10.1145/3778165.

11 Michael Ben-Or, Don Coppersmith, Michael Luby, and Ronitt Rubinfeld. Non-abelian homomorphism testing, and distributions close to their self-convolutions. *Random Struct. Algorithms*, 32(1):49–70, 2008. doi:10.1002/RSA.20182.

12 Eli Ben-Sasson, Madhu Sudan, Salil P. Vadhan, and Avi Wigderson. Randomness-efficient low degree tests and short PCPs via epsilon-biased sets. In *STOC*, pages 612–621, 2003. doi:10.1145/780542.780631.

13 Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Transitive-closure spanners. *SIAM Journal on Computing*, 41(6):1380–1425, 2012. doi:10.1137/110826655.

14 Arnab Bhattacharyya, Swastik Kopparty, Grant Schoenebeck, Madhu Sudan, and David Zuckerman. Optimal testing of Reed-Muller codes. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, Las Vegas, Nevada, USA, October 23-26, 2010*, pages 488–497, 2010. doi:10.1109/FOCS.2010.54.

15 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, 1993. doi:10.1016/0022-0000(93)90044-W.

16 Deeparnab Chakrabarty and C. Seshadhri. Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *STOC*, pages 419–428, 2013. doi:10.1145/2488608.2488661.

17 Irit Dinur and Venkatesan Guruswami. PCPs via low-degree long code and hardness for constrained hypergraph coloring. *Israel Journal of Mathematics*, 209(2):611–649, 2015. doi:10.1007/s11856-015-1231-3.

18 John D Dixon. The probability of generating the symmetric group. *Mathematische Zeitschrift*, 110:199–205, 1969.

19 Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. Improved testing algorithms for monotonicity. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 97–108, 1999. doi:10.1007/978-3-540-48413-4_10.

20 Funda Ergün, Sampath Kannan, Ravi Kumar, Ronitt Rubinfeld, and Mahesh Viswanathan. Spot-checkers. *Journal of Computer and System Sciences*, 60(3):717–751, 2000. doi:10.1006/jcss.1999.1692.

21 Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996. doi:10.1145/226643.226652.

22 Katalin Friedl and Madhu Sudan. Some improvements to total degree tests. In *Third Israel Symposium on Theory of Computing and Systems (ISTCS)*, pages 190–198, 1995. doi:10.1109/ISTCS.1995.377032.

23 Peter Gemmell, Richard J. Lipton, Ronitt Rubinfeld, Madhu Sudan, and Avi Wigderson. Self-testing/correcting for polynomials and for approximate functions. In *STOC*, pages 32–42, 1991. doi:10.1145/103418.103429.

24 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998. doi:10.1145/285055.285060.

25 Oded Goldreich and Dana Ron. On sample-based testers. *ACM Trans. Comput. Theory*, 8(2):7:1–7:54, 2016. doi:10.1145/2898355.

26 William Timothy Gowers and Omid Hatami Varzaneh. Inverse and stability theorems for approximate representations of finite groups. *Matematicheskii Sbornik*, 208(12):70–106, 2017.

27 Elena Grigorescu, Swastik Kopparty, and Madhu Sudan. Local decoding and testing for homomorphisms. In Josep Díaz, Klaus Jansen, José D. P. Rolim, and Uri Zwick, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 9th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2006 and 10th International Workshop on Randomization and Computation, RANDOM 2006, Barcelona, Spain, August 28-30 2006, Proceedings*, volume 4110 of *Lecture Notes in Computer Science*, pages 375–385. Springer, 2006. doi:10.1007/11830924_35.

28 Elad Haramaty, Amir Shpilka, and Madhu Sudan. Optimal testing of multivariate polynomials over small prime fields. *SIAM Journal on Computing*, 42(2):536–562, 2013. doi:10.1137/120879257.

29 Johan Håstad and Avi Wigderson. Simple analysis of graph tests for linearity and PCP. *Random Struct. Algorithms*, 22(2):139–160, 2003. doi:10.1002/rsa.10068.

30 Charanjit S. Jutla, Anindya C. Patthak, Atri Rudra, and David Zuckerman. Testing low-degree polynomials over prime fields. *rsa*, 35(2):163–193, 2009. doi:10.1002/rsa.20262.

31 Iden Kalemaj, Sofya Raskhodnikova, and Nithin Varma. Sublinear-time computation in the presence of online erasures. *Theory of Computing*, 19(1):1–48, 2023. doi:10.4086/TOC.2023.V019A001.

32 Tali Kaufman and Dor Minzer. Improved optimal testing results from global hypercontractivity. *SIAM Journal on Computing*, 54(3):625–663, 2025. doi:10.1137/23M1571022.

33 Tali Kaufman and Dana Ron. Testing polynomials over general fields. *SIAM Journal on Computing*, 36(3):779–802, 2006. doi:10.1137/S0097539704445615.

34 Esty Kelman, Ephraim Linder, and Sofya Raskhodnikova. Online versus offline adversaries in property testing. In Raghu Meka, editor, *16th Innovations in Theoretical Computer Science Conference, ITCS 2025, January 7-10, 2025, Columbia University, New York, NY, USA*, volume 325 of *LIPIcs*, pages 65:1–65:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.ITCS.2025.65.

35 Esty Kelman, Uri Meir, Debanuj Nayak, and Sofya Raskhodnikova. Homomorphism testing with resilience to online manipulations. *CoRR*, abs/2511.23363, 2025. doi:10.48550/arXiv.2511.23363.

36 Esty Kelman, Uri Meir, and Kai Zhe Zheng. Optimal testing of reed-muller codes with an online adversary. In Dana Moshkovitz, editor, *41st Computational Complexity Conference (CCC 2026)*, volume 383 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 31:1–31:26, Dagstuhl, Germany, 2026. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2026.31.

37 Subhash Khot and Kunal Mittal. Biased Linearity Testing in the 1% Regime. In Srikanth Srinivasan, editor, *40th Computational Complexity Conference (CCC 2025)*, volume 339 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:23, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2025.10.

38 Martin W Liebeck and Aner Shalev. The probability of generating a finite simple group. *Geometriae dedicata*, 56:103–113, 1995.

39 Alexander Lubotzky. The expected number of random elements to generate a finite group. *Journal of Algebra*, 257(2):452–459, 2002.

40 Andrea Lucchini. The expected number of random elements to generate a finite group. *Monatshefte für Mathematik*, 181(1):123–142, 2016.

41 Nina E Menezes. *Random generation and chief length of finite groups*. PhD thesis, University of St Andrews, 2013.

42 Dor Minzer and Kai Zhe Zheng. Adversarial low degree testing. In David P. Woodruff, editor, *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024, Alexandria, VA, USA, January 7-10, 2024*, pages 4395–4409. SIAM, 2024. doi:10.1137/1.9781611977912.154.

43 Tushant Mittal and Sourya Roy. Derandomized non-abelian homomorphism testing in low soundness regime. *arXiv preprint arXiv:2405.18998*, 2024. doi:10.48550/arXiv.2405.18998.

44 Tushant Mittal and Sourya Roy. A general framework for low soundness homomorphism testing. In Shubhangi Saraf, editor, *17th Innovations in Theoretical Computer Science Conference (ITCS 2026)*, volume 362 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 103:1–103:18, Dagstuhl, Germany, 2026. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2026.103.

45 Cristopher Moore and Alexander Russell. Approximate representations, approximate homomorphisms, and low-dimensional embeddings of groups. *SIAM Journal on Discrete Mathematics*, 29(1):182–197, 2015. doi:10.1137/140958578.

46 Dana Moshkovitz and Ran Raz. Sub-constant error low degree test of almost-linear size. *SIAM Journal on Computing*, 38(1):140–180, 2008. doi:10.1137/060656838.

47 Carl Pomerance. The expected number of random elements to generate a finite abelian group. *Periodica Mathematica Hungarica*, 43(1):191–198, 2002. doi:10.1023/A\%3A1015250102792.

48 Sofya Raskhodnikova. Monotonicity testing. *Masters Thesis, MIT*, 1999.

49 Sofya Raskhodnikova. Testing if an array is sorted. In *Encyclopedia of Algorithms*, pages 2219–2222. Springer, New York, NY, 2016. doi:10.1007/978-1-4939-2864-4_700.

50 Sofya Raskhodnikova and Ronitt Rubinfeld. Linearity testing/testing Hadamard codes. In *Encyclopedia of Algorithms*, pages 1107–1110. Springer, 2016. doi:10.1007/978-1-4939-2864-4_202.

51 Ran Raz and Shmuel Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *STOC*, pages 475–484, 1997. doi:10.1145/258533.258641.

52 Noga Ron-Zewi and Madhu Sudan. A new upper bound on the query complexity of testing generalized Reed-Muller codes. *Theory Comput.*, 9:783–807, 2013. doi:10.4086/toc.2013.v009a025.

53 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996. doi:10.1137/S0097539793255151.

54 Alex Samorodnitsky. Low-degree tests at large distances. In *STOC*, pages 506–515, 2007. doi:10.1145/1250790.1250864.

55 Alex Samorodnitsky and Luca Trevisan. A PCP characterization of NP with optimal amortized query complexity. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 191–199, 2000. doi:10.1145/335305.335329.

56 Alex Samorodnitsky and Luca Trevisan. Gowers uniformity, influence of variables, and PCPs. *SIAM Journal on Computing*, 39(1):323–360, 2009. doi:10.1137/070681612.

57 Miklos Santha and Umesh V Vazirani. Generating quasi-random sequences from semi-random sources. *Journal of computer and system sciences*, 33(1):75–87, 1986. doi:10.1016/0022-0000(86)90044-9.

58 Amir Shpilka and Avi Wigderson. Derandomizing homomorphism testing in general groups. *SIAM Journal on Computing*, 36(4):1215–1230, 2006. doi:10.1137/S009753970444658X.

59 Luca Trevisan. Recycling queries in PCPs and in linearity tests (extended abstract). In *STOC*, pages 299–308, 1998. doi:10.1145/276698.276769.

60 Alek Westover, Edward Yu, and Kai Zhe Zheng. New Direct Sum Tests. In *16th Innovations in Theoretical Computer Science Conference (ITCS 2025)*, volume 325 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 94:1–94:26. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.ITCS.2025.94.

On the LSH Distortion of Ulam and Cayley Similarities

Flavio Chierichetti
Reddit, San Francisco, CA, USA

Mirko Giacchini
Department of Computer Science, Sapienza University of Rome, Italy

Ravi Kumar
Google Research, Mountain View, CA, USA

Erasmo Tani
Department of Computer Science, Sapienza University of Rome, Italy

Abstract

Locality-sensitive hashing (LSH) has found widespread use as a fundamental primitive, particularly to accelerate nearest neighbor search. An LSH scheme for a similarity function $S : \mathcal{X} \times \mathcal{X} \to [0, 1]$ is a distribution over hash functions on $\mathcal{X}$ with the property that the probability of collision of any two elements $x, y \in \mathcal{X}$ is exactly equal to $S(x, y)$. However, not all similarity functions admit exact LSH schemes. The notion of LSH distortion measures how multiplicatively close a similarity function is to having an LSH scheme.

In this work, we study the LSH distortion of the Ulam and Cayley similarities, which are popular similarity measures on permutations of n elements. We show that the Ulam similarity admits a sublinear LSH distortion of $O(n/\sqrt{\log n})$; we also prove a lower bound of $\Omega(n^{0.12})$ on the best LSH distortion achievable. On the other hand, we show that the LSH distortion of the Cayley similarity is $\Theta(n)$.

2012 ACM Subject Classification Theory of computation → Data structures design and analysis; Information systems → Similarity measures; Theory of computation → Lower bounds and information complexity; Theory of computation → Nearest neighbor algorithms

Keywords and phrases Locality-sensitive Hashing, Ulam metric, Cayley Metric, Distortion, Permutations, Representation Theory

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.59

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2605.11921`

Funding *Flavio Chierichetti*: Work done in part while at Sapienza University of Rome. Supported in part by BiCi – Bertinoro international Center for informatics, by a Google Focused Research Award and by the PRIN project 20229BCXNW (funded by the European Union – Next Generation EU, Mission 4 Component 1 CUP B53D23012910006).

Acknowledgements The authors wish to thank Alessandro Panconesi for providing useful ideas for the project.

1 Introduction

Locality-sensitive hashing (LSH) is an important tool for processing high dimensional data [4, 5, 13, 12]. Classic examples of LSH include minHash for the Jaccard similarity [4, 5], simHash for the cosine similarity [6], and sampling hash for the Hamming similarity [12]. LSH has been extensively used for solving (approximate) nearest-neighbor search [1] and similarity search [11, 16, 20]; see `https://www.mit.edu/~andoni/LSH/` for more pointers on classical results.


© Flavio Chierichetti, Mirko Giacchini, Ravi Kumar, and Erasmo Tani;
licensed under Creative Commons License CC-BY 4.0



LSH is a way to represent a similarity S – which in the context of this paper is a bi-variate, reflexive, symmetric function – by a distribution over partitions. These partitions correspond to the pre-images of hash functions, so that two entities are considered similar by a hash function if they both hash to the same value. In other words, an LSH family $\mathcal{H}$ induces a *collision kernel* of the form $S(x, y) = \Pr_{h\sim\mathcal{H}}[h(x) = h(y)]$, for entities x, y in the universe.

Such collision kernels have to satisfy two constraints: they have unit diagonals (capturing reflexivity) and are a convex combination of equivalence-relation matrices (capturing hashing to the same value). Hence, it turns out that many natural and useful similarity measures do not admit an LSH. An interesting line of work studies which similarities have an LSH. Charikar [6] identified a necessary condition for a similarity to admit an LSH: the corresponding distance function (which is often one minus the similarity) must be a metric and must be isometrically embeddable into ℓ_1.

To circumvent this, Chierichetti et al. [9] introduced the notion of LSH distortion, which answers the question: how much is a non-LSHable similarity S multiplicatively close to an LSHable similarity?

Formally, given a similarity S that does not admit an LSH, is there another similarity S' that has an LSH and satisfies:

$$\forall x, y: \quad \frac{1}{\Delta} \cdot S(x, y) \leq S'(x, y) \leq S(x, y)? \tag{1}$$

We call the optimal Δ in (1) the *LSH distortion* of S.

In their paper, Chierichetti et al. showed tight LSH distortion bounds for several set similarities. Note that, since we are considering multiplicative approximations over **similarities**, and we wish to achieve them using collision kernels, the LSH distortion of a similarity can differ significantly from its metric counterpart, and known results for metric distortion do not transfer to this setting.

Our contributions

In this work, we study the LSH distortion of two natural similarities derived from popular metrics on permutations. The first is the normalized Ulam similarity, which measures the length of the longest common subsequence between two permutations. The second is the Cayley similarity, which is the normalized number of cycles in the relative permutation (its distance counterpart is number of transpositions needed to turn one permutation into another).

Our main results are the following:

- We show that the Ulam similarity has sublinear LSH distortion. In particular, we show that it can be approximated by a similarity that has an LSH, with distortion $O(n/\sqrt{\log n})$ (Theorem 5),
- We also show a lower bound of $\Omega(n^{0.12})$ on the LSH distortion of the Ulam similarity (Theorem 10),
- In contrast, we show that the Cayley similarity has an $\Omega(n)$ lower bound on its LSH distortion (Theorem 19), matching a simple $O(n)$ upper bound (Theorem 18).

Techniques

The proof of the upper bound for the Ulam similarity is based on computing a suitable measure of correlation between the given permutation and a random permutation. The collision probability of our scheme depends on how well the input permutations are correlated with a random one. Showing that the collision probability is upper bounded by $\mathrm{LCS}(\cdot, \cdot)/n$ is relatively easy. On the other hand, showing it is also lower bounded by the LCS (up to sublinear distortion) is more challenging.

For the lower bounds on the distortion, we crucially utilize the PSDness of the collision kernel (Lemma 4). The lower bound for the Ulam similarity is based on first constructing a small set of permutations on an 8-element universe that has an LSH distortion of at least 9/7, and then amplifying its cardinality to work for a universe of arbitrarily large size, by using the wreath product on permutations.

The lower bound for the Cayley similarity is based on representation theory, and uses both the fact that every bi-invariant PSD kernel on $\mathbb{S}_n$ can be written as a convex combination of normalized characters, as well as Roichman's character-ratio estimates applied to derangements. The distortion lower bound is obtained by constructing two derangements that share the high-order terms of this decomposition and hence must have close collision probabilities with the identity permutation, but have vastly different Cayley similarities (linear vs constant) to it.

Related work

For permutation metrics, the Ulam metric has been extensively studied through its connection to edit distance and longest common subsequences. Charikar and Krauthgamer [8] proved that the Ulam metric embeds into ℓ_1 with $O(\log n)$ distortion; Andoni and Krauthgamer [3] later proved that embeddings of the Ulam metric into ℓ_1 requires a near-tight distortion of $\Omega(\log n/\log\log n)$. Andoni et al. [2] bypassed the ℓ_1 barrier by embedding Ulam distance with constant distortion into an iterated product metric. Charikar et al. [7] obtained an $O(\log n)$ distortion randomized embedding of Ulam into Hamming space that also carries alignment information. Note that all these previous results are for the Ulam metric; to the best of our knowledge, the corresponding question of Ulam similarity has not been considered in the literature.

Unlike Ulam, the Cayley metric is not hard to embed into ℓ_1. Our work can be viewed as a hardness result for LSHability of the Cayley similarity.

Outline

We begin, in Section 2, by introducing some basic notation. In Section 3, we review the definition of LSH distortion and introduce in detail the central problems of the paper. In Section 4, we provide our $O(n/\sqrt{\log n})$ upper bound on the LSH distortion for the Ulam similarity. We then give a $\Omega(n^{0.12})$ lower bound in Section 5. In Section 6, we give a simple $O(n)$ upper bound on the distortion of the Cayley similarity, that we complement with a matching lower bound in Section 7.

In the appendix of the full version of the paper, we also review some background on the representation theory of $\mathbb{S}_n$, which we rely on for our proofs in Section 7.

2 Notation

Permutations

For a positive integer n, let $[n] := \{1, \ldots, n\}$. Let $\mathbb{S}_n$ be the set of all permutations of $[n]$. We denote by id the *identity* permutation. For a permutation $\pi \in \mathbb{S}_n$ we will sometimes use the one-line notation and sometimes the cycle decomposition.[1] For a permutation

[1] For example, for the permutation $\pi : [5] \to [5]$ such that $\pi(1) = 3$, $\pi(2) = 5$, $\pi(3) = 1$, $\pi(4) = 2$, $\pi(5) = 4$; the one-line representation is $(3, 5, 1, 2, 4)$ and the cycle decomposition is $(1\ 3)(2\ 5\ 4)$.

$\pi \in \mathbb{S}_n$ we denote with $c(\pi)$ the number of *cycles* in its cycle decomposition. Note that $c(\pi) = c(\pi^{-1}) = c(\sigma\pi\sigma^{-1})$ for any $\sigma \in \mathbb{S}_n$. For $\pi \in \mathbb{S}_n$, $a, b \in [n]$ we write $a <_\pi b$ to indicate that a comes before b in the one-line representation of π (alternatively that $\pi^{-1}(a) < \pi^{-1}(b)$). We write $a \leq_\pi b$ to indicate that either $a = b$ or $a <_\pi b$. For $\pi, \sigma \in \mathbb{S}_n$ we denote with $\mathrm{LCS}(\pi, \sigma)$ the length of the *longest common subsequence* between the one-line representations of π and σ. For $\pi \in \mathbb{S}_n$, we denote with $\mathrm{fix}(\pi)$ the number of *fixed points* of π and with $\mathrm{cycle}_2(\pi)$ the number of cycles of length *two* in the cycle decomposition of π. We also let $\mathrm{supp}(\pi) = n - \mathrm{fix}(\pi)$.

Partitions and Compositions

Recall that, for any positive integer n, a partition $\lambda = (\lambda_1, \ldots, \lambda_r)$ of n is a finite sequence of non-increasing positive integers that sum to n. We write $\lambda \vdash n$ to indicate that λ is a partition of n. We denote by (1^n) the partition $(1, \ldots, 1)$ containing n 1s. Given a partition $\lambda = (\lambda_1, \ldots, \lambda_r) \vdash n$ its *transpose* partition (sometimes also called the *conjugate* partition) is the partition $\lambda' \vdash n$ defined by $\lambda' = (\lambda'_1, \ldots, \lambda'_{\lambda_1})$, where:

$$\forall i \in [\lambda_1] : \lambda'_i = \max\{k \mid \lambda_k \geq i\}.$$

A *composition* of n is an ordered partition of n, i.e., an integer vector $\mu = (\mu_1, \ldots, \mu_\ell)$ such that $\mu_i \geq 0$ for every $i \in [\ell]$ and $\sum_{i \in [\ell]} \mu_i = n$.

Matrices

A symmetric matrix $M \in \mathbb{R}^{n \times n}$ is *positive semidefinite* (PSD, written $M \succeq 0$) if for all $x \in \mathbb{R}^n$, we have:

$$x^T M x \geq 0.$$

The *trace* operator is the function that takes in a square matrix $M \in \mathbb{R}^{n \times n}$ and returns $\mathrm{tr}(M) = \sum_{i \in [n]} M(i, i)$. Given two square matrices X, Y, we denote by $\mathrm{diag}(X, Y)$ the block matrix:

$$\mathrm{diag}(X, Y) = \begin{pmatrix} X & \mathbf{0} \\ \mathbf{0} & Y, \end{pmatrix}$$

where $\mathbf{0}$ is the all-zeros matrix.

3 Similarities, LSH, and Distortion

▶ **Definition 1** (Similarity). *Given a set $\mathcal{X}$, a* similarity *(or* similarity function*) on $\mathcal{X}$ is a function $S : \mathcal{X} \times \mathcal{X} \to [0, 1]$, satisfying:*
(a) Reflexivity: *$S(x, x) = 1$ for all $x \in \mathcal{X}$,*
(b) Symmetry: *$S(x, y) = S(y, x)$ for all $x, y \in \mathcal{X}$.*
A similarity space *is simply a pair $(\mathcal{X}, S)$, where $\mathcal{X}$ is a set, and S is a similarity function on $\mathcal{X}$.*

Given a similarity space, it is natural to ask whether one can find a small representation for elements of $\mathcal{X}$ that preserve the values of $S(\cdot, \cdot)$. A natural approach to obtaining this is locality-sensitive hashing, which aims to construct randomized embeddings of $\mathcal{X}$ into discrete (0-1) similarity spaces that preserve $S(\cdot, \cdot)$ in expectation. We recall the key definition here.

▶ **Definition 2** (Locality-Sensitive Hashing (LSH)). *A* locality-sensitive hashing (LSH) *scheme for a similarity space* $(\mathcal{X}, S)$ *is a distribution* $\mathcal{H}$ *over functions from* $\mathcal{X}$ *into some finite set* $\mathcal{Y}$ *with the property that:*

$$\Pr_{h\sim\mathcal{H}}[h(x) = h(y)] = S(x, y),$$

for all $x, y \in \mathcal{X}$.

We say that a similarity function S is *LSHable* if it admits an LSH scheme.

Chierichetti et al. [9] give a notion of distortion for similarity functions, quantifying the degree to which non-LSHable similarities are far from being LSHable. We recall the key definitions.

▶ **Definition 3** ([9]). *We say a distribution* $\mathcal{H}$ *over functions from* $\mathcal{X}$ *to some finite set* $\mathcal{Y}$ Δ-approximates *the space* $(\mathcal{X}, S)$*, if:*

$$\frac{1}{\Delta} \cdot S(x, y) \leq \Pr_{h\sim\mathcal{H}}[h(x) = h(y)] \leq S(x, y),$$

for all $x, y \in \mathcal{X}$*. The LSH* distortion *(or simply distortion) of a similarity space* $(\mathcal{X}, \mathcal{S})$ *is the smallest* $\Delta \geq 1$ *such that there exists a distribution* $\mathcal{H}$ *that* Δ*-approximates* $(\mathcal{X}, S)$.

A simple argument shows that the distortion of a similarity space $(\mathcal{X}, S)$ could be characterized as the smallest $\Delta \geq 1$ such that there exists an LSHable similarity function $S' : \mathcal{X} \times \mathcal{X} \to [0, 1]$ with the property that:

$$\frac{1}{\Delta} \cdot S(x, y) \leq S'(x, y) \leq S(x, y), \tag{2}$$

for all $x, y \in \mathcal{X}$.

The focus of this work is the distortion of two similarity spaces in which $\mathcal{X}$ is the symmetric group $\mathbb{S}_n$: the Ulam similarity and the Cayley similarity.

Ulam Similarity

The *Ulam distance* between two permutations $\pi, \sigma \in \mathbb{S}_n$ is defined as

$$\mathrm{d}^{\mathrm{U}}(\pi, \sigma) := n - \mathrm{LCS}(\pi, \sigma),$$

We define the (normalized) *Ulam similarity* in a natural way:

$$\mathrm{S}^{\mathrm{U}}(\pi, \sigma) = 1 - \frac{n - \mathrm{LCS}(\pi, \sigma)}{n} = \frac{\mathrm{LCS}(\pi, \sigma)}{n}.$$

Cayley Similarity

The *Cayley distance* between two permutations $\pi, \sigma \in \mathbb{S}_n$ is defined as the minimum number of (not necessarily adjacent) transpositions required to transform one permutation into the other, and it is also equivalent to $\mathrm{d}^{\mathrm{C}}(\pi, \sigma) := n - c(\pi^{-1}\sigma) = n - c(\pi\sigma^{-1})$ (see, e.g., [10, page 118]). We define a notion of (normalized) *Cayley similarity* in a natural way:

$$\mathrm{S}^{\mathrm{C}}(\pi, \sigma) = 1 - \frac{\mathrm{d}^{\mathrm{C}}(\pi, \sigma)}{n} = \frac{c(\pi^{-1}\sigma)}{n}.$$

Positive Semi-Definiteness

In order to prove lower bounds on the achievable distortion, we will use the following result. Let $\mathcal{H}$ be a probability distribution over functions from a finite domain $\mathcal{X}$ into a finite range $\mathcal{Y}$. Fixing any ordering of the elements of $\mathcal{X}$, we define the matrix $P_{\mathcal{H}} \in \mathbb{R}^{|\mathcal{X}|\times|\mathcal{X}|}$ such that:

$$P_{\mathcal{H}}(x_1, x_2) := \Pr_{h\sim\mathcal{H}}\left[h(x_1) = h(x_2)\right],$$

for $x_1, x_2 \in \mathcal{X}$.

We now show that this matrix is always PSD.

▶ **Lemma 4.** *For any distribution $\mathcal{H}$ over functions from $\mathcal{X}$ to $\mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are finite sets, the matrix $P_{\mathcal{H}}$ is PSD. That is, for all $\{w_x\}_{x\in\mathcal{X}} \in \mathbb{R}^{|\mathcal{X}|}$, we have:*

$$\sum_{x_1\in\mathcal{X}}\sum_{x_2\in\mathcal{X}} w_{x_1} w_{x_2} P_{\mathcal{H}}(x_1, x_2) \geq 0.$$

Proof. Note that for any $x_1, x_2 \in \mathcal{X}$,

$$P(x_1, x_2) = \Pr_{h\sim\mathcal{H}}[h(x_1) = h(x_2)] = \mathop{\mathbb{E}}_{h\sim\mathcal{H}}\left[\mathbf{1}_{[h(x_1)=h(x_2)]}\right],$$

where $\mathbf{1}_{[.]}$ is the indicator function, and hence:

$$\begin{aligned}\sum_{x_1\in\mathcal{X}}\sum_{x_2\in\mathcal{X}} w_x w_y P(x_1, x_2) &= \sum_{x_1\in\mathcal{X}}\sum_{x_2\in\mathcal{X}} w_x w_y \mathop{\mathbb{E}}_{h\sim\mathcal{H}}\left[\mathbf{1}_{[h(x_1)=h(x_2)]}\right] \\ &= \mathop{\mathbb{E}}_{h\sim\mathcal{H}}\left[\sum_{x_1\in\mathcal{X}}\sum_{x_2\in\mathcal{X}} w_{x_1} w_{x_2} \mathbf{1}_{[h(x_1)=h(x_2)]}\right].\end{aligned}$$

Now, for a fixed $h : \mathcal{X} \to \mathcal{Y}$, we show that the inner sum is non-negative, which in turns completes the proof. We have:

$$\sum_{x_1\in\mathcal{X}}\sum_{x_2\in\mathcal{X}} w_{x_1} w_{x_2} \mathbf{1}_{[h(x_1)=h(x_2)]} = \sum_{y\in\mathcal{Y}}\sum_{\substack{x_1\in\mathcal{X}:\\ h(x_1)=y}}\sum_{\substack{x_2\in\mathcal{X}:\\ h(x_2)=y}} w_{x_1} w_{x_2} = \sum_{y\in\mathcal{Y}}\left(\sum_{\substack{x\in\mathcal{X}:\\ h(x)=y}} w_x\right)^2 \geq 0. \quad ◀$$

4 Upper Bound for Ulam Similarity

In this section, we prove the following result:

▶ **Theorem 5.** *The LSH distortion of $(\mathbb{S}_n, \mathrm{S^U})$ is $O(n/\sqrt{\log n})$.*

For two permutations $\tau, \pi \in \mathbb{S}_n$ and $z \in [n]$, define:

$$\mathrm{Rec}_{\tau,z}(\pi) := \{a \in [n] \mid z \leq_\pi a \text{ and } b <_\tau a \text{ for all } b \text{ such that } z \leq_\pi b <_\pi a\}.$$

Intuitively, $\mathrm{Rec}_{\tau,z}(\pi)$ is the set of new record maxima according to τ obtained while scanning the one-line representation of π starting from z.

For $\tau, \pi \in \mathbb{S}_n$, $a, z \in [n]$ define:

$$h_{\tau,z,a}(\pi) := \begin{cases} 1 & \text{if } a \in \mathrm{Rec}_{\tau,z}(\pi), \\ \pi & \text{otherwise.}\end{cases}$$

Let $\mathcal{H}$ be the uniform distribution over the set of functions: $\{h_{\tau,z,a} \mid \tau \in \mathbb{S}_n, z \in [n], a \in [n]\}$. Recall that for $\pi, \sigma \in \mathbb{S}_n$:

$$P_{\mathcal{H}}(\pi,\sigma) := \Pr_{h_{\tau,z,a}\sim\mathcal{H}}[h_{\tau,z,a}(\pi) = h_{\tau,z,a}(\sigma)].$$

We will show that: $O(\sqrt{\log n}/n) \cdot \mathrm{S}^{\mathrm{U}}(\pi,\sigma) \le P_{\mathcal{H}}(\pi,\sigma) \le \mathrm{S}^{\mathrm{U}}(\pi,\sigma)$.

▶ **Lemma 6.** *For $\pi, \sigma \in \mathbb{S}_n$, $\pi \neq \sigma$, it holds that:*

$$P_{\mathcal{H}}(\pi,\sigma) = \frac{1}{n} \cdot \mathop{\mathbb{E}}_{\tau,z}\left[\left|\operatorname{Rec}_{\tau,z}(\pi) \cap \operatorname{Rec}_{\tau,z}(\sigma)\right|\right].$$

Proof. Note that for $\pi \neq \sigma$, we have that $h_{\tau,z,a}(\pi) = h_{\tau,z,a}(\sigma)$ if and only if $a \in \operatorname{Rec}_{\tau,z}(\pi) \cap \operatorname{Rec}_{\tau,z}(\sigma)$. Conditioned on the choice of τ and z, there are n possible values for a and only $\left|\operatorname{Rec}_{\tau,z}(\pi) \cap \operatorname{Rec}_{\tau,z}(\sigma)\right|$ of them make the hashes of π and σ equal. This concludes the proof. ◀

▶ **Lemma 7.** *For any $\tau, \pi, \sigma \in \mathbb{S}_n$ and $z \in [n]$, it holds that:*

$$1 \le \left|\operatorname{Rec}_{\tau,z}(\pi) \cap \operatorname{Rec}_{\tau,z}(\sigma)\right| \le \mathrm{LCS}(\pi,\sigma).$$

Proof. Since $z \in \operatorname{Rec}_{\tau,z}(\pi) \cap \operatorname{Rec}_{\tau,z}(\sigma)$, the set $\operatorname{Rec}_{\tau,z}(\pi) \cap \operatorname{Rec}_{\tau,z}(\sigma)$ is always non-empty, proving the first inequality.

We now show the second inequality. If there is only one element in the intersection, the result is trivially true, since $\mathrm{LCS}(\pi,\sigma) \ge 1$ for any $\pi, \sigma \in \mathbb{S}_n$. Otherwise, consider any distinct $x, y \in \operatorname{Rec}_{\tau,z}(\pi) \cap \operatorname{Rec}_{\tau,z}(\sigma)$. Suppose without loss of generality that $x <_\pi y$. By definition of $\operatorname{Rec}_{\tau,z}(\pi)$, since y is observed after x in the one-line representation of π, it must hold that $x <_\tau y$. Moreover, we have $x <_\sigma y$, for otherwise we would have $x >_\tau y$, leading to a contradiction.

Therefore, if we let $\operatorname{Rec}_{\tau,z}(\pi) \cap \operatorname{Rec}_{\tau,z}(\sigma) = r_1 <_\pi \cdots <_\pi r_t$, we also have $r_1 <_\sigma \cdots <_\sigma r_t$, meaning that $r_1, \ldots, r_t$ is a common subsequence of π and σ, and therefore $t \le \mathrm{LCS}(\pi,\sigma)$. ◀

Fixed π, σ such that $\mathrm{LCS}(\pi,\sigma) \ge 2$, define $L := \mathrm{LCS}(\pi,\sigma)$ and let $C = (c_1, \ldots, c_L)$ be a longest common subsequence. For $1 \le i < j \le L$, define the intervals:

$$I^{\pi}_{i,j} := \{x \in [n] \mid c_i \le_\pi x \le_\pi c_j\}, \quad \text{and} \quad I^{\sigma}_{i,j} := \{x \in [n] \mid c_i \le_\sigma x \le_\sigma c_j\}.$$

Define also $W_{i,j} := |I^{\pi}_{i,j} \cup I^{\sigma}_{i,j}|$.

▶ **Lemma 8.** *For any $\pi, \sigma \in \mathbb{S}_n$ such that $L := \mathrm{LCS}(\pi,\sigma) \ge 2$, it holds that:*

$$P_{\mathcal{H}}(\pi,\sigma) \ge \frac{1}{n^2} \cdot \sum_{1\le i<j\le L} \frac{1}{W_{i,j}}.$$

Proof. Consider the case $\pi \neq \sigma$, otherwise the result is trivial, since the right-hand side is always less than 1. For $\tau \in \mathbb{S}_n$, and $1 \le i < j \le L$, suppose that:

$$c_j \ge_\tau x \quad \text{for all } x \in I^{\pi}_{i,j} \cup I^{\sigma}_{i,j}.$$

Under this assumption, it holds that: $c_j \in \operatorname{Rec}_{\tau,c_i}(\pi) \cap \operatorname{Rec}_{\tau,c_i}(\sigma)$. Indeed, we have that c_j is the maximum (according to τ) of all indices $I^{\pi}_{i,j}$ between c_i and c_j in the one-line representation of π and therefore it will be added to $\operatorname{Rec}_{\tau,z}(\pi)$ when starting from $z = c_i$ (and the same argument holds for σ).

By Lemma 6 and the remark above, we have:

$$\begin{aligned}
P_{\mathcal{H}}(\pi,\sigma) &= \frac{1}{n}\cdot \mathop{\mathbb{E}}_{\tau,z}\left[\left|\operatorname{Rec}_{\tau,z}(\pi)\cap\operatorname{Rec}_{\tau,z}(\sigma)\right|\right] \\
&\geq \frac{1}{n}\cdot\sum_{i=1}^{L-1}\Pr[z=c_i]\cdot\mathop{\mathbb{E}}_{\tau}\left[\left|\operatorname{Rec}_{\tau,z}(\pi)\cap\operatorname{Rec}_{\tau,z}(\sigma)\right|\,\middle|\, z=c_i\right] \\
&= \frac{1}{n}\cdot\sum_{i=1}^{L-1}\frac{1}{n}\cdot\sum_{j=1}^{n}\Pr_{\tau}[j\in\operatorname{Rec}_{\tau,c_i}(\pi)\cap\operatorname{Rec}_{\tau,c_i}(\sigma)] \\
&\geq \frac{1}{n^2}\sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\Pr_{\tau}[c_j\in\operatorname{Rec}_{\tau,c_i}(\pi)\cap\operatorname{Rec}_{\tau,c_i}(\sigma)] \\
&\geq \frac{1}{n^2}\sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\Pr_{\tau}\left[c_j\geq_{\tau} x \text{ for all } x\in I^{\pi}_{i,j}\cup I^{\sigma}_{i,j}\right] \\
&= \frac{1}{n^2}\sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\frac{1}{W_{i,j}},
\end{aligned}$$

where the last inequality follows from the remark above and the last equality uses that for a uniform at random $\tau\in\mathbb{S}_n$, the element ranked highest among $I^{\pi}_{i,j}\cup I^{\sigma}_{i,j}$ is chosen uniformly at random among its elements. ◀

▶ **Lemma 9.** *For any $\pi,\sigma\in\mathbb{S}_n$ such that $L:=\operatorname{LCS}(\pi,\sigma)\geq 4$, it holds that:*

$$\sum_{1\leq i<j\leq L}\frac{1}{W_{i,j}}\geq\frac{L^2\ln L}{32n}.$$

Proof. For $1<t\leq L$, define:

$$w_t:=\left|I^{\pi}_{t-1,t}\right|+\left|I^{\sigma}_{t-1,t}\right|.$$

For $1\leq i<L$ and $1\leq d\leq L-i$, define $S_{i,d}:=\sum_{t=i+1}^{i+d}w_t$. Note that, for $1\leq i<j\leq L$, we have:

$$W_{i,j}\leq|I^{\pi}_{i,j}|+|I^{\sigma}_{i,j}|\leq\sum_{t=i+1}^{j}w_t=S_{i,(j-i)}. \tag{3}$$

Therefore, we have:

$$\sum_{1\leq i<j\leq L}\frac{1}{W_{i,j}}=\sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\frac{1}{W_{i,j}}\overset{(3)}{\geq}\sum_{i=1}^{L-1}\sum_{j=i+1}^{L}\frac{1}{S_{i,(j-i)}}=\sum_{i=1}^{L-1}\sum_{d=1}^{L-i}\frac{1}{S_{i,d}}=\sum_{d=1}^{L-1}\sum_{i=1}^{L-d}\frac{1}{S_{i,d}}. \tag{4}$$

For a fixed $1\leq d<L$, by the Cauchy–Schwarz inequality, we have:

$$\left(\sum_{i=1}^{L-d}\frac{1}{S_{i,d}}\right)\cdot\left(\sum_{i=1}^{L-d}S_{i,d}\right)\geq\left(\sum_{i=1}^{L-d}\sqrt{\frac{1}{S_{i,d}}}\cdot\sqrt{S_{i,d}}\right)^2=(L-d)^2. \tag{5}$$

Note that each element of $[n]$ can be counted at most twice in w_t and it can appear in at most two w_t's, therefore:

$$\sum_{t=2}^{L}w_t\leq 4n.$$

Moreover, each w_t can appear in at most d distinct $S_{i,d}$ for $i \in \{1, \ldots, L-d\}$, giving:

$$\sum_{i=1}^{L-d} S_{i,d} \leq d \sum_{t=2}^{L} w_t \leq 4 \cdot d \cdot n.$$

Combining this inequality with (5) gives:

$$\sum_{i=1}^{L-d} \frac{1}{S_{i,d}} \geq \frac{(L-d)^2}{\sum_{i=1}^{L-d} S_{i,d}} \geq \frac{(L-d)^2}{4 \cdot d \cdot n}.$$

Plugging this inequality into (4) we obtain:

$$\sum_{1 \leq i < j \leq L} \frac{1}{W_{i,j}} \geq \sum_{d=1}^{L-1} \frac{(L-d)^2}{4 \cdot d \cdot n} \geq \sum_{d=1}^{\lfloor L/2 \rfloor} \frac{(L-d)^2}{4 \cdot d \cdot n} \geq \frac{L^2}{16n} \sum_{d=1}^{\lfloor L/2 \rfloor} \frac{1}{d} \geq \frac{L^2 \ln L}{32n},$$

where the last inequality uses that for $L \geq 4$:

$$\sum_{d=1}^{\lfloor L/2 \rfloor} \frac{1}{d} \geq \ln(\lfloor L/2 \rfloor + 1) \geq \ln\left(\sqrt{L}\right) = \frac{\ln L}{2}.$$ ◀

We are now ready to prove the main result of this section:

Proof of Theorem 5. It is sufficient to show that for each $\pi, \sigma \in \mathbb{S}_n$ it holds that:

$$\Theta(\sqrt{\log n}/n) \cdot \mathrm{S}^{\mathrm{U}}(\pi, \sigma) \leq P_{\mathcal{H}}(\pi, \sigma) \leq \mathrm{S}^{\mathrm{U}}(\pi, \sigma).$$

Note that if $\pi = \sigma$ then $P_{\mathcal{H}}(\pi, \sigma) = \mathrm{S}^{\mathrm{U}}(\pi, \sigma) = 1$. Therefore, consider the case where $\pi \neq \sigma$. By combining Lemma 6 and Lemma 7 we have:

$$\frac{1}{n} \leq P_{\mathcal{H}}(\pi, \sigma) \leq \frac{\mathrm{LCS}(\pi, \sigma)}{n} = \mathrm{S}^{\mathrm{U}}(\pi, \sigma). \tag{6}$$

Therefore, we only have to prove $P_{\mathcal{H}}(\pi, \sigma) \geq \Omega(\sqrt{\log n}/n) \cdot \mathrm{S}^{\mathrm{U}}(\pi, \sigma)$ to conclude the proof. Consider first the case where $\mathrm{LCS}(\pi, \sigma) \leq 3$. In this case we have:

$$P_{\mathcal{H}}(\pi, \sigma) \geq \frac{1}{n} \geq \frac{1}{3} \cdot \frac{\mathrm{LCS}(\pi, \sigma)}{n} = \frac{1}{3} \cdot \mathrm{S}^{\mathrm{U}}(\pi, \sigma).$$

Consider now the case $\mathrm{LCS}(\pi, \sigma) \geq 4$ and denote $L := \mathrm{LCS}(\pi, \sigma)$. By combining Lemma 8 and Lemma 9 we obtain:

$$P_{\mathcal{H}}(\pi, \sigma) \geq \frac{L^2 \ln L}{32 \cdot n^3} = \frac{L \ln L}{32 \cdot n^2} \cdot \mathrm{S}^{\mathrm{U}}(\pi, \sigma). \tag{7}$$

By (6) we also have:

$$P_{\mathcal{H}}(\pi, \sigma) \geq \frac{1}{L} \cdot \mathrm{S}^{\mathrm{U}}(\pi, \sigma). \tag{8}$$

We split the case $\mathrm{LCS}(\pi, \sigma) \geq 4$ in two sub-cases. Suppose first that $\mathrm{LCS}(\pi, \sigma) \leq \frac{n}{\sqrt{\ln n}}$. Then, (8) gives: $P_{\mathcal{H}}(\pi, \sigma) \geq \sqrt{\ln n}/n \cdot \mathrm{S}^{\mathrm{U}}(\pi, \sigma)$. Suppose instead that $\mathrm{LCS}(\pi, \sigma) > \frac{n}{\sqrt{\ln n}}$. Then, by (7) we have:

$$P_{\mathcal{H}}(\pi, \sigma) \geq \frac{\frac{n}{\sqrt{\ln n}} \ln\left(\frac{n}{\sqrt{\ln n}}\right)}{32n^2} \cdot \mathrm{S}^{\mathrm{U}}(\pi, \sigma) = \left(\frac{\ln n - \frac{1}{2} \ln \ln n}{32 \cdot \sqrt{\ln n} \cdot n}\right) \cdot \mathrm{S}^{\mathrm{U}}(\pi, \sigma) \geq \frac{\sqrt{\ln n}}{64 \cdot n} \cdot \mathrm{S}^{\mathrm{U}}(\pi, \sigma),$$

where we used that $\frac{\ln n}{2} \geq \frac{\ln \ln n}{2}$ for $n \geq 3$. ◀

5 Lower Bound for Ulam Similarity

In this section, we prove the following result.

▶ **Theorem 10.** *Let n be any integer of the form 8^{2^k}, for integer $k \geq 0$. The LSH distortion of $(\mathbb{S}_n, \mathrm{S^U})$ is $\Omega(n^\delta)$, where $\delta \geq 0.12$.*

At a high level, we obtain a lower bound on the LSH distortion by starting with a small instance that exhibits constant distortion, and then iteratively amplifying it to handle larger values of n. In each step, both the distortion bound and the permutation domain grow quadratically. Our base case has $n = 8$, therefore our construction produces instances for any n of the form 8^{2^k}, where k is a non-negative integer. The main tool enabling this amplification is the wreath product.

For any two subsets $A, B \subseteq \mathbb{S}_n$, we define the similarity matrix $M_{A,B} \in \mathbb{R}^{|A|\times|B|}$ by $M_{A,B}(\pi, \sigma) = \mathrm{S^U}(\pi, \sigma)$ for any $\pi \in A$ and $\sigma \in B$. For simplicity, when A and B are the same, we will adopt a lighter notation, and denote by M_A the matrix $M_{A,A}$. We note that the matrix $M_{(A\cup B)}$ has a simple block structure:

$$M_{(A\cup B)} = \begin{pmatrix} M_A & M_{A,B} \\ M_{B,A} & M_B \end{pmatrix},$$

where we fixed an ordering in which the permutations in A come before those in B.

▶ **Definition 11** (Witness). *For $n > 0$, let $A, B \subseteq \mathbb{S}_n$ be disjoint subsets such that $|A| = |B|$ and $M_A = M_B$. Let $W \in \mathbb{R}^{(|A|+|B|)\times(|A|+|B|)}$ be a PSD block matrix such that:*

$$W = \begin{pmatrix} U & -V \\ -V^T & U \end{pmatrix} \succeq 0,$$

where $U, V \in \mathbb{R}^{|A|\times|A|}$ are entry-wise non-negative symmetric matrices such that $\mathrm{tr}(U \cdot M_A) \neq 0$. We call the triple (A, B, W) a witness *of size n and we define its value to be:*

$$\nu(A, B, W) := \frac{\mathrm{tr}(V \cdot M_{A,B})}{\mathrm{tr}(U \cdot M_A)}.$$

We now show that the existence of a witness implies a lower bound to the LSH distortion for Ulam similarity on $\mathbb{S}_n$.

▶ **Lemma 12.** *Let (A, B, W) be a witness of size n (Definition 11). Then, the LSH distortion for Ulam similarity space $(\mathbb{S}_n, \mathrm{S^U})$ is at least the value of the witness, $\nu(A, B, W)$, even when restricted only to permutations in $A \cup B \subseteq \mathbb{S}_n$.*

Proof. Consider any distribution $\mathcal{H}$ over functions on $\mathbb{S}_n$ which Δ-approximates $(A \cup B, \mathrm{S^U})$ for $\Delta \geq 1$. For every $\pi, \sigma \in A \cup B$ we have:

$$\frac{M_{A\cup B}(\pi, \sigma)}{\Delta} = \frac{\mathrm{S^U}(\pi, \sigma)}{\Delta} \leq P_{\mathcal{H}}(\pi, \sigma) \leq \mathrm{S^U}(\pi, \sigma) = M_{A\cup B}(\pi, \sigma). \tag{9}$$

Note that, having fixed an ordering of the elements of $A \cup B$ that puts the elements of A before the elements of B, $P_{\mathcal{H}}$ gets naturally divided into four square blocks:

$$P_{\mathcal{H}} = \begin{pmatrix} P_A & P_{A,B} \\ P_{B,A} & P_B \end{pmatrix}.$$

By Lemma 4, $P_{\mathcal{H}} \succeq 0$ and $W \succeq 0$, we have $\mathrm{tr}(WP_{\mathcal{H}}) \geq 0$. Expanding blockwise, and using $V^T = V$ and $P_{B,A}^T = P_{A,B}$, we have:

$$\mathrm{tr}(UP_A) + \mathrm{tr}(UP_B) - 2\mathrm{tr}(VP_{A,B}) \geq 0.$$

Note that, by Equation (9), $P_{\mathcal{H}}(\pi,\sigma) \leq M_{A\cup B}(\pi,\sigma)$ and $P_{\mathcal{H}}(\pi,\sigma) \geq (1/\Delta)\cdot M_{A\cup B}(\pi,\sigma)$ for all π,σ. Therefore, we have:

$$\mathrm{tr}(U\cdot M_A) + \mathrm{tr}(U\cdot M_B) \geq \frac{2}{\Delta}\cdot \mathrm{tr}(V\cdot M_{A,B}).$$

Since $M_A = M_B$, we have $\mathrm{tr}(U\cdot M_A) = \mathrm{tr}(U\cdot M_B)$. Hence,

$$\Delta \geq \frac{\mathrm{tr}(V\cdot M_{A,B})}{\mathrm{tr}(U\cdot M_A)} = \nu(A,B,W). \qquad \blacktriangleleft$$

For permutations $\pi,\sigma \in \mathbb{S}_n$ the *wreath product* $\pi \wr \sigma \in \mathbb{S}_{n^2}$ is defined as follows. The elements of $[n^2]$ are partitioned into n blocks where the ith block, $i \in [n]$, contains elements $\Sigma_i := \{(i-1)\cdot n+1,\ldots,i\cdot n\}$. The resulting permutation $\pi\wr\sigma$ is obtained by permuting the elements in each block with the inner permutation σ and then permuting the blocks with the outer permutation π. For two sets $A,B \subseteq \mathbb{S}_n$, we define $A \wr B := \{\pi\wr\sigma \mid \pi\in A, \sigma\in B\}$. We use the following key property of the wreath product for amplification:

▶ **Lemma 13.** *For any $\pi,\sigma,\tau_1,\tau_2 \in \mathbb{S}_n$ we have:*

$$\mathrm{LCS}(\pi\wr\sigma, \tau_1\wr\tau_2) = \mathrm{LCS}(\pi,\tau_1)\cdot\mathrm{LCS}(\sigma,\tau_2).$$

Proof. In the permutation $\pi\wr\sigma$, the blocks appear in the order given by π, and within every block the elements appear in the order given by σ; similarly for $\tau_1\wr\tau_2$.

We first prove that $\mathrm{LCS}(\pi\wr\sigma,\tau_1\wr\tau_2) \leq \mathrm{LCS}(\pi,\tau_1)\cdot\mathrm{LCS}(\sigma,\tau_2)$. Let Z be any common subsequence of $\pi\wr\sigma$ and $\tau_1\wr\tau_2$. Since the alphabets Σ_i are pairwise disjoint, an element from a block Σ_i can only be matched with the same element from the same block Σ_i. Therefore, if we track the sequence of block indices that contribute at least one matched element to Z, then this is a common subsequence of the outer permutations π and τ_1; thus, the number of contributing blocks is at most $\mathrm{LCS}(\pi,\tau_1)$.

Let Σ_i be any contributing block. The matched elements of Z in this block is a common subsequence of the ordering of Σ_i inside $\pi\wr\sigma$ and the ordering of Σ_i inside $\tau_1\wr\tau_2$. These two internal orderings are σ and τ_2, relabeled with the alphabet Σ_i; therefore, this block contributes at most $\mathrm{LCS}(\sigma,\tau_2)$ matched elements. Summing over all contributing blocks, we obtain $\mathrm{LCS}(\pi\wr\sigma,\tau_1\wr\tau_2) \leq \mathrm{LCS}(\pi,\tau_1)\cdot\mathrm{LCS}(\sigma,\tau_2)$.

We now show $\mathrm{LCS}(\pi\wr\sigma,\tau_1\wr\tau_2) \geq \mathrm{LCS}(\pi,\tau_1)\cdot\mathrm{LCS}(\sigma,\tau_2)$. Consider an optimal common subsequence of the outer permutations π and τ_1, consisting of $\mathrm{LCS}(\pi,\tau_1)$ block indices. For each such matched block Σ_i, consider an optimal common subsequence of the inner permutations σ and τ_2, relabeled using Σ_i. Since the block indices appear in the same relative order in both π and τ_1, concatenating these common subsequences over the chosen blocks yields a valid common subsequence of $\pi\wr\sigma$ and $\tau_1\wr\tau_2$ of length $\mathrm{LCS}(\pi,\tau_1)\cdot\mathrm{LCS}(\sigma,\tau_2)$. $\blacktriangleleft$

We now show how to use the wreath product to amplify the size of the witness. We will make use of the *Kronecker product*. We recall that for two matrices $X \in \mathbb{R}^{n\times m}$ and $Y \in \mathbb{R}^{s\times t}$, the Kronecker product $X\otimes Y \in \mathbb{R}^{ns\times mt}$ is the following block matrix:

$$X\otimes Y = \begin{pmatrix} x_{11}\cdot Y & \ldots & x_{1m}\cdot Y \\ \vdots & \ddots & \vdots \\ x_{n1}\cdot Y & \ldots & x_{nm}\cdot Y \end{pmatrix}.$$

▶ **Lemma 14.** *Consider any disjoint sets $A, B \subseteq \mathbb{S}_n$ with $|A| = |B|$. Let $A' = A \wr A$ and $B' = B \wr B$. Then, we have:*

$$M_{(A' \cup B')} = \begin{pmatrix} M_A \otimes M_A & M_{A,B} \otimes M_{A,B} \\ M_{B,A} \otimes M_{B,A} & M_B \otimes M_B \end{pmatrix}.$$

Proof. Let $\pi, \sigma \in A' \cup B'$ and suppose that $\pi = \pi_1 \wr \pi_2$, $\sigma = \sigma_1 \wr \sigma_2$. By Lemma 13,

$$\begin{aligned} M_{(A' \cup B')}(\pi, \sigma) &= \frac{\mathrm{LCS}(\pi, \sigma)}{n^2} = \frac{\mathrm{LCS}(\pi_1, \sigma_1)}{n} \cdot \frac{\mathrm{LCS}(\pi_2, \sigma_2)}{n} \\ &= M_{(A \cup B)}(\pi_1, \sigma_1) \cdot M_{(A \cup B)}(\pi_2, \sigma_2). \end{aligned}$$

Then, by considering the ordering over A' obtained by the ordering of A when we consider first the outer permutation and then the inner permutation (and similarly for B'), we have that each entry of the matrix is equivalent to the Kronecker product. ◀

▶ **Lemma 15.** *Let (A, B, W) be a witness of size n. Let $A' = A \wr A$, $B' = B \wr B$, and:*

$$W' = \begin{pmatrix} U \otimes U & -(V \otimes V) \\ -(V^T \otimes V^T) & U \otimes U \end{pmatrix}.$$

Then, (A', B', W') is a witness of size n^2 and $\nu(A', B', W') \geq \nu(A, B, W)^2$.

Proof. Note that $U \otimes U$ and $V \otimes V$ are non-negative and symmetric. Moreover, since $W \succeq 0$, we have that $W \otimes W \succeq 0$. Define:

$$W'' = \begin{pmatrix} U \otimes U & V \otimes V \\ V^T \otimes V^T & U \otimes U \end{pmatrix}.$$

Since W'' is a principal sub-matrix of $W \otimes W$, we have that $W'' \succeq 0$. Observe now that $W' = \mathrm{diag}(I, -I) \cdot W'' \cdot \mathrm{diag}(I, -I)$, and therefore $W' \succeq 0$ since for any matrix $X \succeq 0$ and any matrix Y, $YXY^T \succeq 0$. Moreover, by Lemma 14 and since $M_A = M_B$, we also have $M_{A'} = M_{B'}$. Then, (A', B', W') is a valid witness of size n^2. We now compute its value.

For matrices A, B, C, D it holds that $\mathrm{tr}((A \otimes B)(C \otimes D)) = \mathrm{tr}(AC) \cdot \mathrm{tr}(BD)$. By using this property, and by Lemma 14, we have:

$$\begin{aligned} \mathrm{tr}((V \otimes V) \cdot M_{A',B'}) &= \mathrm{tr}((V \otimes V) \cdot (M_{A,B} \otimes M_{A,B})) = \mathrm{tr}(V \cdot M_{A,B})^2, \text{ and} \\ \mathrm{tr}((U \otimes U) \cdot M_{A'}) &= \mathrm{tr}((U \otimes U) \cdot (M_A \otimes M_A)) = \mathrm{tr}(U \cdot M_A)^2. \end{aligned}$$

Note that since $\mathrm{tr}(U \cdot M_A) \neq 0$, $\mathrm{tr}((U \otimes U) M_{A'}) \neq 0$. Then:

$$\nu(A', B', W') = \frac{\mathrm{tr}((V \otimes V) \cdot M_{A',B'})}{\mathrm{tr}((U \otimes U) \cdot M_{A'})} = \left(\frac{\mathrm{tr}(V \cdot M_{A,B})}{\mathrm{tr}(U \cdot M_A)} \right)^2 = \nu(A, B, W)^2. \qquad \blacktriangleleft$$

We now provide our starting instance.

▶ **Proposition 16.** *There exists a witness (A_0, B_0, W_0) of size 8 and value $\geq 9/7$.*

Proof. Let $A_0 = \{A_1, \ldots, A_4\}$ and $B_0 = \{B_1, \ldots, B_4\}$, where $\tau = (1\,2\,7\,8)(3\,6\,5\,4) = (2, 7, 6, 3, 4, 5, 8, 1)$ and $B_i = \tau A_i$:

$$\begin{aligned} A_1 &= (1, 2, 3, 4, 5, 6, 7, 8), & B_1 &= (2, 7, 6, 3, 4, 5, 8, 1), \\ A_2 &= (4, 3, 2, 1, 8, 7, 6, 5), & B_2 &= (3, 6, 7, 2, 1, 8, 5, 4), \\ A_3 &= (6, 5, 8, 7, 2, 1, 4, 3), & B_3 &= (5, 4, 1, 8, 7, 2, 3, 6), \\ A_4 &= (7, 8, 5, 6, 3, 4, 1, 2), & B_4 &= (8, 1, 4, 5, 6, 3, 2, 7). \end{aligned}$$

Since B_0 is obtained from A_0 by relabeling, we have $M_{A_0} = M_{B_0}$. A direct computation gives

$$M_{A_0} = \frac{1}{8}\begin{pmatrix} 8 & 2 & 2 & 2 \\ 2 & 8 & 2 & 2 \\ 2 & 2 & 8 & 2 \\ 2 & 2 & 2 & 8 \end{pmatrix}, \qquad M_{A_0,B_0} = \frac{1}{8}\begin{pmatrix} 5 & 4 & 4 & 5 \\ 4 & 5 & 5 & 4 \\ 4 & 5 & 5 & 4 \\ 5 & 4 & 4 & 5 \end{pmatrix}.$$

The matrix W_0 for this instance is given by $U = V = J_4$ (where J_4 is the all-ones 4×4 matrix). First,

$$W = \begin{pmatrix} U & -V \\ -V^T & U \end{pmatrix} = \begin{pmatrix} J_4 & -J_4 \\ -J_4 & J_4 \end{pmatrix} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \otimes J_4 \succeq 0.$$

Next, it is easy to calculate

$$\mathrm{tr}(V \cdot M_{A_0,B_0}) = \frac{1}{8}(18 \cdot 4) = 9, \qquad \mathrm{tr}(U \cdot M_{A_0}) = \frac{1}{8}(14 \cdot 4) = 7.$$

Hence, $\nu(A_0, B_0, W_0) = 9/7$. ◀

▶ **Lemma 17.** *There exists a constant $\delta > 0$ such that, for any $n_0 > 0$, there exists $n \geq n_0$ such that the LSH distortion of the Ulam similarity S^{U} on $\mathbb{S}_n$ is at least $\Omega(n^\delta)$.*

Proof. Let $\Delta_0 = 9/7$ and $m = 8$. Let k be the smallest integer such that $m^{2^k} \geq n_0$, we set $n = m^{2^k}$. Note that $k = \log_2 \log_m(n)$.

Starting from the witness of Proposition 16 and applying Lemma 15 k times we obtain a witness of size $m^{2^k} = n$ and value $\geq \Delta_0^{2^k}$. By Lemma 12, this implies that the LSH distortion on $\mathbb{S}_n$ is at least:

$$\Delta_0^{2^k} = \Delta_0^{\log_m(n)} = n^{\log_m \Delta_0} \geq n^{0.120856} = \Omega(n^\delta),$$

for $\delta = 0.12$. ◀

Note that Lemma 17 directly implies Theorem 10. The proof is elementary.

6 Upper Bound for Cayley Similarity

In this section, we show that the LSH distortion for Cayley similarity is no more than n.

▶ **Theorem 18.** *The LSH distortion of $(\mathbb{S}_n, \mathrm{S}^{\mathrm{C}})$ is at most n.*

Proof. Let $\mathcal{H}$ be the uniform at random distribution over the functions $h : \mathbb{S}_n \to [n]$. Clearly, for each $\pi, \sigma \in \mathbb{S}_n$, $\pi \neq \sigma$:

$$\Pr_{h \sim \mathcal{H}}[h(\pi) = h(\sigma)] = \frac{1}{n}.$$

Since $\frac{1}{n} \leq \mathrm{S}^{\mathrm{C}}(\pi, \sigma) \leq 1$ for all $\pi, \sigma \in \mathbb{S}_n$, we have:

$$\frac{1}{n} \cdot \mathrm{S}^{\mathrm{C}}(\pi, \sigma) \leq \Pr_{h \sim \mathcal{H}}[h(\pi) = h(\sigma)] \leq \mathrm{S}^{\mathrm{C}}(\pi, \sigma).$$ ◀

7 Lower Bound for Cayley Similarity

In this section, we show the following result:

▶ **Theorem 19.** *The LSH distortion of* $(\mathbb{S}_n, \mathrm{S}^{\mathrm{C}})$ *is* $\Omega(n)$.

The proof will rely on some standard results in the representation theory of $\mathbb{S}_n$. For convenience, we review these standard results in the appendix of the full version of the paper (see also, e.g., [19, 14, 17]).

We begin with the following definition:

▶ **Definition 20** (Bi-invariance). *We say that a function* $f : \mathbb{S}_n \times \mathbb{S}_n \to \mathbb{R}$ *is* bi-invariant *if:*

$$f(\pi, \sigma) = f(\tau_1 \pi \tau_2, \tau_1 \sigma \tau_2),$$

for any $\pi, \sigma, \tau_1, \tau_2 \in \mathbb{S}_n$. *We say a bi-invariant function is a* bi-invariant PSD kernel *if, in addition, it satisfies:*

$$\sum_{\pi,\sigma \in \mathbb{S}_n} a_\sigma a_\pi f(\pi, \sigma) \geq 0,$$

for every $a \in \mathbb{R}^{\mathbb{S}_n}$. *We also require that* $f(\mathrm{id}, \mathrm{id}) = 1$ *for* f *to be a PSD kernel.*

Recall that a *class function* on $\mathbb{S}_n$ is a function that is constant on each conjugacy class of $\mathbb{S}_n$. We recall the following property of bi-invariant functions. (Note that equivalent results are known in the literature, and hold for more general groups – see, e.g., [14, Chapter 4.5] – but we include a proof anyway for completeness.)

▶ **Lemma 21.** *A function* $f : \mathbb{S}_n \times \mathbb{S}_n \to \mathbb{R}$ *is bi-invariant if and only if there exists a class function* $g : \mathbb{S}_n \to \mathbb{R}$ *such that:*

$$f(\sigma, \pi) = g(\sigma \pi^{-1}),$$

for every $\sigma, \pi \in \mathbb{S}_n$.

Proof. Let $f : \mathbb{S}_n \times \mathbb{S}_n \to \mathbb{R}$ be a bi-invariant function. Define $g : \mathbb{S}_n \to \mathbb{R}$ as:

$$g(\pi) := f(\pi, \mathrm{id}).$$

Then we have:

$$f(\pi, \sigma) = f(\pi\sigma^{-1}, \sigma\sigma^{-1}) = f(\pi\sigma^{-1}, \mathrm{id}) = g(\pi\sigma^{-1})$$

for any $\pi, \sigma \in \mathbb{S}_n$. Moreover, g is a class function, since, for any $\pi, \sigma \in \mathbb{S}_n$:

$$g(\sigma\pi\sigma^{-1}) = f(\sigma\pi\sigma^{-1}, \mathrm{id}) = f(\pi, \sigma^{-1}\sigma) = f(\pi, \mathrm{id}) = g(\pi),$$

where the second equality follows from the bi-invariance of f.

On the other hand, suppose that $f : \mathbb{S}_n \times \mathbb{S}_n \to \mathbb{R}$ is a function such that $f(\pi, \sigma) = g(\pi\sigma^{-1})$ for some class function g, then, for any $\pi, \sigma, \tau \in \mathbb{S}_n$:

$$f(\pi\tau, \sigma\tau) = g(\pi\tau\tau^{-1}\sigma^{-1}) = g(\pi\sigma^{-1}) = f(\pi, \sigma),$$

and:

$$f(\tau\pi, \tau\sigma) = g(\tau\pi\sigma^{-1}\tau^{-1}) = g(\pi\sigma^{-1}) = f(\pi, \sigma),$$

where the second equality uses the fact that g is a class function. Hence f is bi-invariant. ◀

A crucial property of the Cayley similarity is that it is bi-invariant, as we now show.

▶ **Lemma 22.** $\mathrm{S}^{\mathrm{C}}(\cdot,\cdot)$ *is bi-invariant.*

Proof. We have that, for any choice of π and σ, $\mathrm{S}^{\mathrm{C}}(\pi,\sigma) = \frac{c(\pi\sigma^{-1})}{n}$. Since any two conjugate permutations have the same cycle type, the function $\pi \mapsto \frac{c(\pi)}{n}$ is a class function, and the result follows from Lemma 21. ◀

▶ **Definition 23** (Bi-invariant Distribution over Functions)**.** *We will say that a distribution $\mathcal{H}$ over functions from $\mathbb{S}_n$ into some set $\mathcal{Y}$ is* bi-invariant *if the function $P_{\mathcal{H}}(\cdot,\cdot)$ is bi-invariant.*

We now show that when studying the minimum achievable distortion for $(\mathbb{S}_n, \mathrm{S}^{\mathrm{C}})$, any LSH family can be assumed to be bi-invariant.

▶ **Lemma 24.** *Suppose there exists a distribution $\mathcal{H}$ over functions from $\mathbb{S}_n$ into a finite set $\mathcal{Y}$ that Δ-approximates $(\mathbb{S}_n, \mathrm{S}^{\mathrm{C}})$, then there exists a bi-invariant distribution $\mathcal{H}'$ that Δ-approximates $(\mathbb{S}_n, \mathrm{S}^{\mathrm{C}})$.*

Proof. We construct $\mathcal{H}'$ by pre-composing the elements of $\mathcal{H}$ with random left and right permutations: define a function $h' \sim \mathcal{H}'$ as $h'(\pi) = h(\alpha\pi\beta)$, where h is drawn from $\mathcal{H}$ and α, β are drawn uniformly and independently from $\mathbb{S}_n$. We first show that $\mathcal{H}'$ Δ-approximates $(\mathbb{S}_n, \mathrm{S}^{\mathrm{C}})$. For every choice of $\pi, \sigma \in \mathbb{S}_n$, we have:

$$\begin{aligned}\Pr_{h'\sim\mathcal{H}'}[h'(\pi) = h'(\sigma)] &= \mathop{\mathbb{E}}_{\alpha,\beta\sim\mathbb{S}_n}\left[\Pr_{h\sim\mathcal{H}}[h(\alpha\pi\beta) = h(\alpha\sigma\beta)]\right] \\ &\leq \mathop{\mathbb{E}}_{\alpha,\beta\sim\mathbb{S}_n}\left[\mathrm{S}^{\mathrm{C}}(\alpha\pi\beta, \alpha\sigma\beta)\right] \\ &\overset{\text{Lemma 22}}{=} \mathop{\mathbb{E}}_{\alpha,\beta\sim\mathbb{S}_n}\left[\mathrm{S}^{\mathrm{C}}(\pi,\sigma)\right] = \mathrm{S}^{\mathrm{C}}(\pi,\sigma),\end{aligned}$$

where the first inequality uses that $\mathcal{H}$ Δ-approximates $(\mathbb{S}_n, \mathrm{S}^{\mathrm{C}})$. Similarly:

$$\begin{aligned}\Pr_{h'\sim\mathcal{H}'}[h'(\pi) = h'(\sigma)] &= \mathop{\mathbb{E}}_{\alpha,\beta\sim\mathbb{S}_n}\left[\Pr_{h\sim\mathcal{H}}[h(\alpha\pi\beta) = h(\alpha\sigma\beta)]\right] \\ &\geq \mathop{\mathbb{E}}_{\alpha,\beta\sim\mathbb{S}_n}\left[\frac{1}{\Delta}\cdot\mathrm{S}^{\mathrm{C}}(\alpha\pi\beta, \alpha\sigma\beta)\right] \\ &\overset{\text{Lemma 22}}{=} \mathop{\mathbb{E}}_{\alpha,\beta\sim\mathbb{S}_n}\left[\frac{1}{\Delta}\cdot\mathrm{S}^{\mathrm{C}}(\pi,\sigma)\right] = \frac{1}{\Delta}\mathrm{S}^{\mathrm{C}}(\pi,\sigma),\end{aligned}$$

Showing that indeed $\mathcal{H}'$ Δ-approximates $(\mathbb{S}_n, \mathrm{S}^{\mathrm{C}})$. Finally, we verify that $P_{\mathcal{H}'}$ is bi-invariant. For any $\pi, \sigma \in \mathbb{S}_n$ and any left multiplier τ_1 and right multiplier τ_2:

$$P_{\mathcal{H}'}(\tau_1\pi\tau_2, \tau_1\sigma\tau_2) = \mathbb{E}_{\alpha,\beta\sim\mathbb{S}_n}[P_{\mathcal{H}}(\alpha\tau_1\pi\tau_2\beta, \alpha\tau_1\sigma\tau_2\beta)].$$

Because α and β are drawn uniformly from the group, the products $\alpha' = \alpha\tau_1$ and $\beta' = \tau_2\beta$ are also uniformly distributed. Thus, the expectation is identical to $\mathbb{E}_{\alpha',\beta'}[P_{\mathcal{H}}(\alpha'\pi\beta', \alpha'\sigma\beta')] = P_{\mathcal{H}'}(\pi,\sigma)$, completing the proof. ◀

Note also that, by Lemma 4, $P_{\mathcal{H}'}$ is PSD and it is therefore a bi-invariant PSD kernel.

Recall that each partition $\lambda \vdash n$ is associated with a distinct irreducible representation S^λ of $\mathbb{S}_n$, we denote by $\chi_\lambda : \mathbb{S}_n \to \mathbb{C}$ its character, and by $d_\lambda \in \mathbb{N}$ its dimension.

▶ **Lemma 25.** *Any bi-invariant PSD kernel P on $\mathbb{S}_n$ can be written as a convex combination of normalized characters of irreducible representations of $\mathbb{S}_n$ as follows:*

$$\forall \sigma, \pi \in \mathbb{S}_n : \quad P(\sigma, \pi) = \sum_{\lambda : \lambda \vdash n} w_\lambda \frac{\chi_\lambda(\sigma\pi^{-1})}{d_\lambda},$$

where $w_\lambda \geq 0$ and $\sum_\lambda w_\lambda = 1$.

Proof. Let $\phi : \mathbb{S}_n \to \mathbb{R}$ be the class function satisfying:

$$\phi(\sigma\pi^{-1}) = P(\sigma, \pi),$$

the existence of which is guaranteed by Lemma 21. By a result of Kondor [14, Corollary 4.5.5[2]], since ϕ is PSD, we have:

$$\phi = \sum_{\lambda \vdash n} a_\lambda \cdot \chi_\lambda, \tag{10}$$

for some choice of $\{a_\lambda\}_{\lambda \vdash n}$, where each $a_\lambda \in \mathbb{R}_{\geq 0}$.

In particular:

$$1 = P(\mathrm{id}, \mathrm{id}) = \phi(\mathrm{id}) = \sum_{\lambda \vdash n} a_\lambda \cdot \chi_\lambda(\mathrm{id}) = \sum_{\lambda \vdash n} a_\lambda \cdot d_\lambda,$$

where we use that $\chi_\lambda(\mathrm{id}) = d_\lambda$. Letting $w_\lambda := a_\lambda \cdot d_\lambda$, together with Equation (10) completes the proof. ◀

For our asymptotic bounds, it is necessary to isolate the trivial representation (n), where $\chi_{(n)}(\cdot) = 1$ and $d_{(n)} = 1$ and the sign representation (1^n), where $\chi_{(1^n)}(\sigma) = \mathrm{sgn}(\sigma)$ and $d_{(1^n)} = 1$. Thus, it is helpful to write

$$P(\mathrm{id}, \sigma) = w_0 + w_{\mathrm{sgn}} \cdot \mathrm{sgn}(\sigma) + \sum_{\lambda \neq (n), (1^n)} w_\lambda \frac{\chi_\lambda(\sigma)}{d_\lambda}. \tag{11}$$

For a permutation $\sigma \in \mathbb{S}_n$, the *fixed-point* count is defined as $\mathrm{fix}(\sigma) = |\{i \in [n] \mid \sigma(i) = i\}|$, i.e., the number of length-1 cycles. A *derangement* is a permutation σ that moves all elements, i.e., $\mathrm{fix}(\sigma) = 0$. For any $\sigma \in \mathbb{S}_n$, we also define the value $\mathrm{supp}(\sigma)$ as $n - \mathrm{fix}(\sigma)$. We recall the following result.

▶ **Theorem 26** ([18])**.** *There exist absolute constants $\varphi > 0$ and $0 < q < 1$ such that for any partition $\lambda = (\lambda_1, \ldots, \lambda_k) \vdash n$, any $n \geq 4$ and any permutation $\sigma \in \mathbb{S}_n$:*

$$\left| \frac{\chi_\lambda(\sigma)}{d_\lambda} \right| \leq \left(\max \left\{ \frac{\lambda_1}{n}, \frac{k}{n}, q \right\} \right)^{\varphi \cdot \mathrm{supp}(\sigma)}.$$

Theorem 26 allows us to prove the following.

▶ **Lemma 27.** *There exist absolute constants $\varphi > 0$ and $0 < q < 1$ such that for any $n \geq 4$, any partition $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_k) \vdash n$, and any derangement $\sigma \in \mathbb{S}_n$,*

$$\left| \frac{\chi_\lambda(\sigma)}{d_\lambda} \right| \leq e^{-\varphi \cdot \alpha},$$

where $\alpha = \min\{n - \lambda_1, n - k, (1 - q)n\}$.

[2] We remark that Kondor's result is stated in terms of what he refers to as "Positive Definite" kernels, which are, in reality, positive semidefinite kernels, as he himself explains earlier in the thesis.

Proof. Since σ is a derangement, $\mathrm{fix}(\sigma) = 0$; therefore $\mathrm{supp}(\sigma) = n - \mathrm{fix}(\sigma) = n$. Furthermore, from the definition of α, we have:

$$\max\left\{\frac{\lambda_1}{n}, \frac{k}{n}, q\right\} = 1 - \frac{\alpha}{n}.$$

Substituting these into the bound from Theorem 26 yields:

$$\left|\frac{\chi_\lambda(\sigma)}{d_\lambda}\right| \leq \left(1 - \frac{\alpha}{n}\right)^{\varphi n} \leq e^{-\varphi\cdot\alpha}.$$ ◀

▶ **Lemma 28** (Theorem 1.2 (i) from [15], paraphrased)**.** *For any derangement $\sigma \in \mathbb{S}_n$ and any partition $\lambda \vdash n$:*

$$\left|\frac{\chi_\lambda(\sigma)}{d_\lambda}\right| \leq \frac{1}{d_\lambda^{1/2-o(1)}}.$$

▶ **Lemma 29.** *For any partition $\lambda = (\lambda_1, \ldots, \lambda_k) \vdash n$ with $\lambda_1 \geq \frac{11}{12}\cdot n$, it holds that: $d_\lambda \geq \left(\frac{n}{2(n-\lambda_1)}\right)^{n-\lambda_1}$*

Proof. Let $\gamma = n - \lambda_1$. By the hook length formula (see the appendix of the full version of the paper), we have:

$$\begin{aligned}
d_\lambda = \frac{n!}{\prod h(i,j)} &\geq \frac{n!}{(n-2\gamma)!\cdot n^\gamma\cdot\gamma^\gamma} \\
&= \frac{n\cdots(n-2\gamma+1)}{n^\gamma\gamma^\gamma} \\
&= \frac{n}{n}\cdots\frac{n-\gamma+1}{n}\cdot\frac{(n-\gamma)\cdots(n-2\gamma+1)}{\gamma^\gamma} \\
&\geq \left(\frac{n-\gamma+1}{n}\right)^\gamma\cdot\left(\frac{n-2\gamma+1}{\gamma}\right)^\gamma \\
&\geq \left(\frac{n^2-6\cdot n\cdot\gamma}{\gamma\cdot n}\right)^\gamma \\
&= \left(\frac{n}{\gamma}-6\right)^\gamma \\
&= \left(\frac{n}{\gamma}\left(1-\frac{6\gamma}{n}\right)\right)^\gamma \\
&\geq \left(\frac{n}{2\gamma}\right)^\gamma,
\end{aligned}$$

where the last inequality used the fact that $\gamma \leq \frac{n}{12}$. ◀

▶ **Lemma 30.** *Consider any sufficiently large n, any derangement $\sigma \in \mathbb{S}_n$, and let P be a bi-invariant PSD kernel. Let:*

$$P(\tau,\pi) = \sum_{\lambda\vdash n} w_\lambda \frac{\chi_\lambda(\tau\pi^{-1})}{d_\lambda},$$

be the decomposition guaranteed by Lemma 25, then:

$$\left|\sum_{\substack{\lambda\vdash n\\ \lambda\neq(n),(1^n)}} w_\lambda\frac{\chi_\lambda(\sigma)}{d_\lambda}\right| \leq \frac{C}{n},$$

for some constant C.

Proof. We have:

$$\left|\sum_{\substack{\lambda\vdash n\\ \lambda\neq(n),(1^n)}} w_\lambda \frac{\chi_\lambda(\sigma)}{d_\lambda}\right| \leq \max_{\substack{\lambda\vdash n\\ \lambda\neq(n),(1^n)}} \left|\frac{\chi_\lambda(\sigma)}{d_\lambda}\right| \sum_{\substack{\lambda\vdash n\\ \lambda\neq(n),(1^n)}} w_\lambda \leq \max_{\substack{\lambda\vdash n\\ \lambda\neq(n),(1^n)}} \left|\frac{\chi_\lambda(\sigma)}{d_\lambda}\right|,$$

where we first apply Hölder's inequality on ℓ_∞ and ℓ_1 and then we use the fact that $\sum_{\lambda\vdash n} w_\lambda = 1$, as well as the non-negativity of the w_λs. We now give an upper bound for the right-hand side. Let φ and q be the constants given by Lemma 27. Let $\lambda = (\lambda_1, \ldots, \lambda_k) \vdash n$, and let $\alpha = \min\{n-\lambda_1, n-k, (1-q)n\}$. Since $\lambda \neq (n), (1^n)$, we have $\min\{n-\lambda_1, n-k\} \geq 1$. We obtain an upper bound by considering five possible cases for the value of α.

Case 1: $\alpha = (1-q)n$. Using Lemma 27, we have:

$$\left|\frac{\chi_\lambda(\sigma)}{d_\lambda}\right| \leq e^{-\varphi\cdot\alpha} = e^{-\Theta(n)} = o\left(\frac{1}{n}\right).$$

For the rest of the proof we will assume that $\alpha = n - \lambda_1 \geq 1$. This is without loss of generality, since, for any partition that does not satisfy this condition (i.e., for which $\alpha = n - k$), its transpose partition does satisfy it, and the values of $\left|\frac{\chi_\lambda(\sigma)}{d_\lambda}\right|$ are the same for the two partitions.

Case 2: $\alpha \geq \frac{1}{\varphi}\ln n$. Using Lemma 27 we have:

$$\left|\frac{\chi_\lambda(\sigma)}{d_\lambda}\right| \leq e^{-\varphi\cdot\alpha} \leq \frac{1}{n}.$$

Case 3: $3 \leq \alpha < \frac{1}{\varphi}\ln n$. Using Lemma 28 and Lemma 29 and recalling that $\alpha = n - \lambda_1$, and that therefore $\lambda_1 = n - \alpha > n - \frac{1}{\varphi}\ln n \geq \frac{11}{12}n$ for sufficiently large n, we obtain:

$$\left|\frac{\chi_\lambda(\sigma)}{d_\lambda}\right| \leq \left(\frac{1}{d_\lambda}\right)^{1/2-o(1)} \leq \left(\frac{2\alpha}{n}\right)^{\alpha\left(\frac{1}{2}-o(1)\right)} \leq \left(\frac{(2/\varphi)\cdot\ln(n)}{n}\right)^{3/2-o(1)} = O\left(\frac{1}{n}\right),$$

where we use that $\alpha \geq 3$ and $\alpha < O(\log n)$.

Case 4: $\alpha = 1$. In this case, the only partition is $\lambda = (n-1, 1)$, corresponding to the standard representation, which has dimension $n-1$. Its character is known to satisfy $\chi_\lambda(\pi) = \mathrm{fix}(\pi) - 1$ (See [19, Example 2.3.8]). Since σ is a derangement, $\mathrm{fix}(\sigma) = 0$, and the character $\chi_\lambda(\sigma) = -1$. We then have:

$$\left|\frac{\chi_\lambda(\sigma)}{d_\lambda}\right| = \left|\frac{-1}{n-1}\right| = O\left(\frac{1}{n}\right).$$

Case 5: $\alpha = 2$. In this case, the only two partitions are $\lambda = (n-2, 2)$ and $\mu = (n-2, 1, 1)$. By (19) in Example 32 in the appendix of the full version of the paper, we can write the character of the permutation module $M^{(n-2,2)}$ as the sum of irreducible characters as follows:

$$\chi_{M^{(n-2,2)}} = \chi_{(n-2,2)} + \chi_{(n-1,1)} + \chi_{(n)}.$$

By equation (17) in the appendix of the full version of the paper, this equation becomes:

$$\mathrm{cycle}_2(\pi) + \binom{\mathrm{fix}(\pi)}{2} = \chi_{(n-2,2)}(\pi) + \mathrm{fix}(\pi) - 1 + 1,$$

for all $\pi \in \mathbb{S}_n$, where $\text{cycle}_2(\pi)$ is the number of length-two cycles in the cycle decomposition of π. And hence:

$$\chi_{(n-2,2)}(\pi) = \text{cycle}_2(\pi) + \binom{\text{fix}(\pi)}{2} - \text{fix}(\pi),$$

for all $\pi \in \mathbb{S}_n$.
Similarly, by Equation (20) in Example 32 in the appendix of the full version of the paper, we can write the character of the permutation module $M^{(n-2,1,1)}$ as:

$$\chi_{M^{(n-2,1,1)}} = \chi_{(n-2,1,1)} + \chi_{(n-2,2)} + 2\chi_{(n-1,1)} + \chi_{(n)}.$$

By Equation (18) in that same appendix, this equation becomes:

$$\text{fix}(\pi)(\text{fix}(\pi) - 1) = \chi_{(n-2,1,1)} + \chi_{(n-2,2)} + 2(\text{fix}(\pi) - 1) + 1$$

for all $\pi \in \mathbb{S}_n$, which gives:

$$\chi_{(n-2,1,1)} = 1 - \text{cycle}_2(\pi) + \frac{\text{fix}(\pi)(\text{fix}(\pi) - 3)}{2}$$

for all $\pi \in \mathbb{S}_n$. Since σ is a derangement and $\text{fix}(\sigma) = 0$, we have:

$$\chi_{(n-2,2)}(\sigma) = \text{cycle}_2(\sigma),$$

and:

$$\chi_{(n-2,1,1)}(\sigma) = 1 - \text{cycle}_2(\sigma).$$

By Example 21 in the appendix of the full version of the paper: $d_\lambda = n(n-3)/2$ and $d_\mu = (n-1)(n-2)/2$. Putting things together, we have that:

$$\left|\frac{\chi_\lambda(\sigma)}{d_\lambda}\right| = \frac{2 \cdot \text{cycle}_2(\sigma)}{n(n-3)} \leq \frac{n}{n(n-3)} = \frac{1}{n-3} = O\left(\frac{1}{n}\right),$$

and:

$$\left|\frac{\chi_\mu(\sigma)}{d_\mu}\right| = \frac{2 \cdot (\text{cycle}_2(\sigma) - 1)}{(n-1)(n-2)} \leq \frac{n}{(n-1)(n-2)} = O\left(\frac{1}{n}\right).$$

This completes the proof of Lemma 30. ◀

We can now complete the proof of Theorem 19.

Proof of Theorem 19. Let $n \geq 6$ be a large enough integer so that Lemma 30 holds. Let $\mathcal{H}$ be a distribution that Δ-approximates $(\mathbb{S}_n, \mathrm{S}^{\mathrm{C}})$. Without loss of generality, by Lemma 24, we can assume that $P_{\mathcal{H}}$ is a bi-invariant PSD kernel.

We let σ^* be a derangement with $\Theta(n)$ cycles and π^* be a derangement with $O(1)$ cycles and having the same sign as σ^*. We explicitly define these permutations expressing them in their cycle decomposition. If n is even, we let:

$$\sigma^* := \prod_{i=1}^{n/2} (2i-1, 2i),$$

and observe that $c(\sigma^*) = n/2$. If instead n is odd, we define:

$$\sigma^* := (1,2,3) \prod_{i=2}^{\frac{n-1}{2}} (2i, 2i+1),$$

and observe that $c(\sigma^*) = (n-1)/2$ and σ^* can be decomposed in $(n+1)/2$ transpositions. Now, if n is even and $\operatorname{sgn}(\sigma^*) = 1$ or if n is odd and $\operatorname{sgn}(\sigma^*) = -1$, we define:

$$\pi^* = (1, 2, \ldots, n-3)(n-2, n-1, n),$$

and note that $\operatorname{sgn}(\pi^*) = \operatorname{sgn}(\sigma^*)$ since it can be decomposed into $n-2$ transpositions. If instead n is even and $\operatorname{sgn}(\sigma^*) = -1$ or if n is odd and $\operatorname{sgn}(\sigma^*) = 1$, we define:

$$\pi^* = (1, 2, \ldots, n),$$

and note that $\operatorname{sgn}(\pi^*) = \operatorname{sgn}(\sigma^*)$ since it can be decomposed into $n-1$ transpositions.

Observe that, since $n \geq 6$:

$$\mathrm{S}^{\mathrm{C}}(\mathrm{id}, \sigma^*) \geq \frac{(n-1)/2}{n} \geq \frac{1}{3} \qquad \text{and} \qquad \mathrm{S}^{\mathrm{C}}(\mathrm{id}, \pi^*) \leq \frac{2}{n}.$$

Since $\mathcal{H}$ Δ-approximates $(\mathbb{S}_n, \mathrm{S}^{\mathrm{C}})$, we have:

$$P_{\mathcal{H}}(\mathrm{id}, \pi^*) \leq \mathrm{S}^{\mathrm{C}}(\mathrm{id}, \pi^*) \leq \frac{2}{n}.$$

Plugging this into (11), and letting C be the constant in the statement of Lemma 30:

$$w_0 + w_{\mathrm{sgn}} - \frac{C}{n} \overset{\text{Lemma 30}}{\leq} w_0 + w_{\mathrm{sgn}} \cdot \operatorname{sgn}(\pi^*) + \sum_{\substack{\lambda \vdash n \\ \lambda \neq (n), (1^n)}} w_\lambda \frac{\chi_\lambda(\pi^*)}{d_\lambda} \overset{(11)}{=} P_{\mathcal{H}}(\mathrm{id}, \pi^*) \leq \frac{2}{n},$$

giving:

$$w_0 + w_{\mathrm{sgn}} \leq \frac{2 + C}{n}. \tag{12}$$

Again applying Equation (11) and Lemma 30, we have:

$$P_{\mathcal{H}}(\mathrm{id}, \sigma^*) \overset{(11),\text{Lemma 30}}{\leq} w_0 + w_{\mathrm{sgn}} + \frac{C}{n} \overset{(12)}{\leq} \frac{2 + 2C}{n}.$$

On the other hand:

$$P_{\mathcal{H}}(\mathrm{id}, \sigma^*) \geq \frac{1}{\Delta} \mathrm{S}^{\mathrm{C}}(\mathrm{id}, \sigma^*) = \frac{1}{3\Delta}.$$

Combining the last two equations yields:

$$\Delta \geq \frac{n}{6(1 + C)},$$

completing the proof. ◀

8 Conclusions and Open Questions

Our work shows that the Ulam similarity enjoys sublinear LSH distortion whereas the Cayley similarity does not. An obvious open problem is to close the glaring yet tantalizing gap between the upper bound $(n/\sqrt{\log n})$ and the lower bound $(\Omega(n^{0.12}))$ for the Ulam similarity; our current belief is that the correct LSH distortion bound is $\tilde{\Theta}(\sqrt{n})$. Improving the base case (Proposition 16) is a natural direction, though there are computational limitations to finding a larger and better base instance. While it seems possible to improve our upper bound to $\tilde{O}(n/\log n)$, going beyond this to $\tilde{O}(n^{1-\Omega(1)})$ might require an entirely new approach. Extending our results to other permutation similarities such as reversal is another interesting direction; our current lower bound techniques seem inadequate for the block reversal similarity, for which we conjecture an $\Omega(n)$ lower bound.

LLM Usage

The Ulam upper bound was obtained after weeks of interaction with and help from an LLM; the LLM-generated proof was hand-verified and somewhat simplified. The solver for the base case of the Ulam lower bound was written by an LLM; the instance itself was hand-verified. The high-level approach of using representation theory, in particular, Roichman's Theorem and derangements, to obtain the Cayley lower bound was suggested by an LLM (after the initial attempts to extend Ulam lower bound proof approach to Cayley failed). All the final proofs were written and verified by the authors.

References

1 Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *CACM*, 51(1):117–122, 2008. doi:10.1145/1327452.1327494.

2 Alexandr Andoni, Piotr Indyk, and Robert Krauthgamer. Overcoming the l_1 non-embeddability barrier: algorithms for product metrics. In *SODA*, pages 865–874, 2009. doi:10.1137/1.9781611973068.94.

3 Alexandr Andoni and Robert Krauthgamer. The computational hardness of estimating edit distance. *SICOMP*, 39(6):2398–2429, 2010. doi:10.1137/080716530.

4 Andrei Z Broder. On the resemblance and containment of documents. In *SEQUENCES*, pages 21–29, 1997. doi:10.1109/SEQUEN.1997.666900.

5 Andrei Z. Broder, Moses Charikar, Alan M. Frieze, and Michael Mitzenmacher. Min-wise independent permutations. *JCSS*, 60(3):630–659, 2000. doi:10.1006/JCSS.1999.1690.

6 Moses Charikar. Similarity estimation techniques from rounding algorithms. In *STOC*, pages 380–388, 2002. doi:10.1145/509907.509965.

7 Moses Charikar, Ofir Geri, Michael P. Kim, and William Kuszmaul. On estimating edit distance: Alignment, dimension reduction, and embeddings. In *ICALP*, pages 34:1–34:14, 2018. doi:10.4230/LIPIcs.ICALP.2018.34.

8 Moses Charikar and Robert Krauthgamer. Embedding the Ulam metric into l_1. *Theory Comput.*, 2(11):207–224, 2006. doi:10.4086/TOC.2006.V002A011.

9 Flavio Chierichetti, Ravi Kumar, Alessandro Panconesi, and Erisa Terolli. On the distortion of locality sensitive hashing. *SICOMP*, 48(2):350–372, 2019. doi:10.1137/17M1127752.

10 Persi Diaconis. *Group Representations in Probability and Statistics*. Inst. Math. Stat., 1988.

11 Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *VLDB*, pages 518–529, 1999. URL: http://www.vldb.org/conf/1999/P49.pdf.

12 Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *STOC*, pages 604–613, 1998. doi:10.1145/276698.276876.

13 Piotr Indyk, Rajeev Motwani, Prabhakar Raghavan, and Santosh S. Vempala. Locality-preserving hashing in multidimensional spaces. In *STOC*, pages 618–625, 1997. doi:10.1145/258533.258656.

14 Imre Risi Kondor. *Group Theoretical Methods in Machine Learning*. Columbia U., 2008.

15 Michael Larsen and Aner Shalev. Characters of symmetric groups: sharp bounds and applications. *Invent. Math.*, 174(3):645–687, 2008.

16 Qin Lv, William Josephson, Zhe Wang, Moses Charikar, and Kai Li. Multi-probe LSH: efficient indexing for high-dimensional similarity search. In *VLDB*, pages 950–961, 2007. URL: http://www.vldb.org/conf/2007/papers/research/p950-lv.pdf.

17 Paul Renteln. A combinatorial proof of a symmetric group character involution. *The American Mathematical Monthly*, 130(9):855–858, 2023. doi:10.1080/00029890.2023.2242042.

18 Yuval Roichman. Upper bound on the characters of the symmetric groups. *Invent. Math.*, 125:451–485, 1996.

19 Bruce Sagan. *The Symmetric Group: Representations, Combinatorial Algorithms, and Symmetric Functions*. Springer Science & Business Media, 2001.

20 Venu Satuluri and Srinivasan Parthasarathy. Bayesian locality sensitive hashing for fast similarity search. *Proc. VLDB Endow.*, 5(5):430–441, 2012. doi:10.14778/2140436.2140440.

Fine-Grained Complexity via Quantum Natural Proofs

Yanlin Chen
University of Maryland (QuICS), College Park, MD, USA

Yilei Chen
Shanghai Qi Zhi Institute, Tsinghua University, Beijing, China

Rajendra Kumar
Indian Institute of Technology Delhi, New Delhi, India

Subhasree Patro
Eindhoven University of Technology, The Netherlands

Florian Speelman
University of Amsterdam and QuSoft, The Netherlands

Abstract

Buhrman, Patro, and Speelman [11] presented a framework of conjectures that together form a quantum analogue of the strong exponential-time hypothesis and its variants. They called it the QSETH framework. In this paper, using a notion of quantum natural proofs (built from natural proofs introduced by Razborov and Rudich), we show how part of the QSETH conjecture that requires properties to be "compression oblivious" can in many cases be replaced by assuming the existence of quantum-secure pseudorandom functions, a standard hardness assumption. Combined with techniques from Fourier analysis of Boolean functions, we show that properties such as PARITY and MAJORITY are compression oblivious for certain circuit class Λ if subexponentially secure quantum pseudorandom functions exist in Λ, answering an open question in [11].

2012 ACM Subject Classification Theory of computation → Quantum computation theory

Keywords and phrases Fine-grained complexity, Quantum natural proofs, QSETH, Pseudorandom functions

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.60

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/pdf/2504.10363` [13]

Funding *Yanlin Chen*: Acknowledges the support from a QuICS Hartree Postdoctoral Fellowship.
Yilei Chen: Supported by Tsinghua University startup funding, and Shanghai Qi Zhi Institute Innovation Program SQZ202405.
Rajendra Kumar: Supported by the Chandrukha New Faculty Fellowship from IIT Delhi and the Prime Minister Early Career Research Grant from the Anusandhan National Research Foundation (ANRF).
Subhasree Patro: Acknowledges the support from the Dutch Ministry of Education, Culture, and Science through Gravitation project "Challenges in Cyber Security - 024.006.037" for this work.
Florian Speelman: Supported by the Dutch Ministry of Economic Affairs and Climate Policy (EZK), as part of the Quantum Delta NL program, and the project Divide and Quantum "D&Q" NWA.1389.20.241 of the program "NWA-ORC", which is partly funded by the Dutch Research Council (NWO).



© Yanlin Chen, Yilei Chen, Rajendra Kumar, Subhasree Patro, and Florian Speelman;
licensed under Creative Commons License CC-BY 4.0

1 Introduction

A well-studied problem in complexity theory is CNFSAT, the problem of whether a formula, input in conjunctive normal form, has a satisfying assignment. Not only is there no polynomial time known for CNFSAT, there is also no known algorithm that significantly improves over the brute-force method of checking all 2^n assignments. Impagliazzo, Paturi and Zane in [23, 24] conjectured that determining whether an input CNF formula is satisfiable or not cannot be done in time $O(2^{n(1-\varepsilon)})$ for any constant $\varepsilon > 0$. They called it *Strong Exponential-Time Hypothesis* (SETH). As a consequence, SETH-based lower bounds were shown for many computational problems; for example, see [31, 8, 21].

Variants of classical SETH

Subsequent works also explored alternate directions in which the strong exponential-time hypothesis can be weakened and thereby made more plausible. For, e.g.,

1. The *satisfiability* problem is studied for other (more general) succinct representations of Boolean functions such as BPSAT or `Circuit`SAT - the problem of whether a polynomial-size branching program or a polynomial-size circuit has a satisfying assignment. Furthermore, respective hardness conjectures for these satisfiability problems, namely NCSETH and `Circuit`SETH (under the name "`Circuit`SAT form of SETH") have also been discussed together with their implications [2, 30].
2. Another direction of study that has made SETH more plausible is by computing more complicated properties of the formula. For example, counting the number of satisfying assignments and computing whether the number of satisfying assignments is even or odd is captured by #SETH and $\oplus$SETH, respectively; see [17] for example.

Unfortunately, neither SETH nor NCSETH or `Circuit`SETH, hold relative to *quantum* computation, as using Grover's algorithm for unstructured search [19] it is possible to solve the respective satisfiability problems in $O(2^{n/2})$ time - leading to conjectured lower bounds of complexity $\Omega(2^{n/2})$. While these lower bounds have interesting consequences (as shown by [1, 11]), in the quantum case, generalizing the properties to be computed (as suggested in Item 2) is a way to come up with variants of the hypothesis that enable stronger bounds: for many of such tasks it is likely that the quadratic quantum speedup, as given by Grover's algorithm, no longer exists. In fact, motivated by this exact observation, Buhrman, Patro, and Speelman [11] introduced a framework of Quantum Strong Exponential-Time Hypotheses (QSETH) as quantum analogues to SETH, with a striking feature that allows one to technically unify quantum analogues of SETH, $\oplus$SETH, #SETH, NCSETH, $\oplus$NCSETH, etc., all under one umbrella conjecture.

The QSETH framework

Buhrman *et al.* consider the problem in which one is given a formula or a circuit representation of a Boolean function $f : \{0,1\}^n \to \{0,1\}$ and is asked whether a property $\mathrm{P} : \{0,1\}^{2^n} \to \{0,1\}$ on its truth table[1] evaluates to 1. They conjectured that when the circuit representation is *obfuscated* enough (that is, the description of the circuit provides no advantages over treating the circuit as a black-box with query access to its truth table), then for *most*

[1] Truth table of a formula/circuit ϕ on n variables, denoted by $\mathtt{tt}(\phi)$, is a 2^n bit string derived in the following way $\mathtt{tt}(\phi) = \bigcirc_{a\in\{0,1\}^n} \phi(a)$; the symbol $\circ$ denotes concatenation.

properties P (that are "compression oblivious" with respect to the representation), the time taken to compute P on the truth table of poly(n)-sized circuits is lower bounded by $\mathsf{Q}(\mathrm{P})$, i.e., the 1/3-bounded error quantum query complexity of P, on all bit strings of length 2^n.

Note that there are two requirements to using the QSETH framework. Firstly, the input formula or circuit must be "obfuscated enough" - as we already saw this translates to saying that knowing the description of the input must not help towards computing P any more than just having query access to the truth table of the input. The second requirement (which is the main topic of discussion of this paper) is that the property P itself must be "compression oblivious" - a notion introduced in [11] that we discuss next.

Compression-oblivious properties

Buhrman *et al.* defined this notion to capture properties whose *query complexity* on all strings is a lower bound for the *time complexity* to compute the property even for inputs that are "compressible" as a truth table of small formulas (or circuits). This notion is useful (to their framework) because if the formula (or circuit) is obfuscated enough then one can use the quantum query lower bound of such a property to conjecture a time lower bound of computing that property on a small formula (or circuit).[2] Therefore, before using their framework, one has to be convinced that the property is indeed compression-oblivious with respect to the circuit representation considered. They give various results to initiate the study of the set of compression-oblivious languages; see Section 2.3 in [11]. For example, they show the following.

- For simple properties, such as OR, AND, they were able to prove that these are compression-oblivious for polynomial-size AC^0_2 circuits.
- They gave an example of a property that is *not* compression oblivious for polynomial-sized circuits (or any smaller class of representations).[3]
- For more complicated properties, such as PARITY or MAJORITY, they show that proving compression-obliviousness would separate P from PSPACE; see Theorem 9 in [11]. Their result indicates that *unconditionally* proving compression-obliviousness for a complicated property is hard. One way around this would be to directly *conjecture* the compression-obliviousness of such properties, which was the choice of [11]. However, it would be desirable to put this on more solid ground.

[2] Here is an example to illustrate why obfuscation is important. Consider the OR property. The quantum query complexity of OR when restricted to truth tables of small CNF or DNF formulas defined on n input variables is $\Omega(\sqrt{N})$ where $N = 2^n$ - let us denote this statement by $(\star)$. Furthermore, the structure of CNF formulas are not known to help in computing OR more efficiently than $O(\sqrt{N})$. Therefore, we can use the same lower bound from $(\star)$ to conjecture the lower bound of $\Omega(\sqrt{N})$ for CNFSAT in the quantum setting. However, DNF formulas are not obfuscated enough for OR as we can compute OR in poly(n) time when input formulas are DNFs. This means we cannot use the query lower bound of $\sqrt{N}$ from $(\star)$ for the DNFs as we do in the case of CNF formulas.

[3] Consider the following property (Example 7 in [11]), defined on some string $z \in \{0,1\}^{2^n}$, which we view as the truth table of a formula or circuit:

$$\mathrm{P}_{\mathtt{large}-c}(z) = \textsc{parity}(z) \wedge [\text{there exists no circuit } C \text{ of size less than } 2^{n/100} \text{ s.t. } z = \mathtt{tt}(C)].$$

We use $\mathtt{tt}(C)$ to denote the truth table of C. Because most strings are not a truth table of a small circuit, the query complexity of this property is close to the query complexity of PARITY, i.e., $\mathsf{Q}(\mathrm{P}_{\mathtt{large}-c}) = \Omega(2^n)$. Nevertheless, the property is always 0 when restricted to truth tables of small circuits, and therefore trivial to compute. Therefore, $\mathrm{P}_{\mathtt{large}-c}$ is not compression oblivious for polynomial-sized circuits (or any smaller class of representations).

One might ask – *why is the notion of compression-oblivious properties useful for the QSETH framework when it is hard to prove results about this notion for many properties?*

Towards answering this question, consider the $\oplus$QSETH hypothesis - implicitly stated and discussed in [11] and explicitly stated in [14, 12]. It says,

▶ **Conjecture 1** (Conjecture 8.9 in [12]). *For each constant $\delta > 0$, there exists $c > 0$ such that there is no bounded-error quantum algorithm that solves $\oplus$CNFSAT (even restricted to formulas with at most $c \cdot n^2$ clauses) in $O(2^{n(1-\delta)})$ time.*

One can go about refuting this conjecture in two (immediate) ways.

1. Either one could use the description of the input formulas to help deduce PARITY of the truth table corresponding to this formula (which would mean that the CNF formula is not obfuscated enough and one could use the structure of the formula towards computing PARITY more efficiently), or
2. One could use the fact that the set of truth tables on which PARITY is to be computed is "not too big" (as the number of polynomial length formulas are not too many) and potentially use this information to speed up the computation quantumly.

While both approaches seem like good starting points, the likelihood of Item 1 not working out is probably for the same reason why $\oplus$SETH is not refuted yet. Which brings us to Item 2. There is a stark contrast between the possible speedups that quantum can offer over classical computation. For example, there exists a quantum *query* algorithm that can compute *any* property in $\widetilde{O}(\sqrt{N})$ many queries, when the input is promised to be the N-length truth table of a small formula [4, 5, 25]; such an algorithm is impossible in the classical setting. Although efficient in the number of queries, the algorithm itself is not *time* efficient. But the existence of this algorithm (in contrast to the lack of one in the classical setting) does alert us that query complexity cannot naively be used to lower bound time complexity in our setting, without taking care. (The results of the current work will indicate that such a fast algorithm is unlikely to refute $\oplus$QSETH via Item 2.)

As pointed out by Buhrman *et al.* [11], the proof barrier does not allow one to (easily) prove compression-obliviousness of properties such as PARITY or MAJORITY. It is therefore natural to ask if there are other reasons why certain properties, say PARITY for example, should be compression oblivious. The authors leave the following open question:

"What complexity-theoretic assumptions are needed to show that, e.g., the PARITY property, is compression oblivious?"

It is precisely this question we answer in this paper. While our answer doesn't give us a way around *conjecturing* the compression obliviousness of these properties, it gives strong evidence supporting the conjecture and the use of the QSETH framework.

1.1 Main idea: arguing compression obliviousness via quantum natural proofs

Following is the answer to the question asked earlier.

▶ **Theorem 2** (Informal restatement of Theorem 34). *The properties PARITY and MAJORITY are compression oblivious for a circuit class* M *unless there are no quantum-secure* M*-constructible* PRF*s (of a particular type).*[4]

In fact, we conclude the same for a larger class of properties (which includes PARITY and MAJORITY). The details are mentioned in the statement of Theorem 34.

[4] For us, the logical statements "A implies B", "if A then B" and "$\neg A$ unless B" are all equivalent.

1.1.1 Natural proofs by Razborov and Rudich

To prove our results, we use the notion of *natural proofs* introduced by Razborov and Rudich [28]. Consider a Boolean function defined on other Boolean functions, or equivalently defined on their truth tables. Razborov and Rudich called a property (of Boolean functions) *natural* if it satisfies certain "constructivity" and "largeness" conditions. Roughly speaking, for their proof, the constructivity condition requires that the property is decidable in $\text{poly}(n)$ time when the 2^n-sized truth table of an n-input Boolean function is given as input. The largeness condition requires that the property holds for a sufficiently large fraction of the set of all Boolean functions. Moreover, they say a property is useful against a complexity class Λ if every sequence of Boolean functions having the property (infinitely often) defines a language outside of Λ. A *natural proof* is a proof that establishes that a certain language lies outside of Λ. This can be formally defined as (following the notation of Chow [16]),

▶ **Definition 3** (Υ-natural property with density δ, Definition 3 in [16]). *Let Υ be a complexity class and let $\delta := (\delta_n)_{n\in\mathbb{N}}$ denote a sequence of positive real numbers. A property $D := (D_n)_{n\in\mathbb{N}}$ is Υ-natural with density $\delta := (\delta_n)_{n\in\mathbb{N}}$ if the following statements hold:*

1. *largeness: $\exists n', \forall n > n'$, $|D_n| \geq 2^{2^n} \cdot \delta_n$, and*
2. *constructivity: the problem of determining whether a Boolean function $f_n \in D_n$, when given as input the truth table of f_n on n variables, is decidable in Υ.*

Note that the largeness is captured by the density function $(\delta_n)_{n\in\mathbb{N}}$ and constructivity is captured by the complexity class Υ. Additionally, this Υ-natural property is useful against a complexity class Λ if the following statement is satisfied.

▶ **Definition 4** (Useful property, Definition 4 in [16]). *Let Λ denote a complexity class. A property $D := (D_n)$ is useful against complexity class Λ if $\forall (f_n)$, we have $(f_n) \notin \Lambda$ whenever (f_n) is a sequence of Boolean functions satisfying $f_n \in D_n$ for infinitely many n.*

Using these definitions, Razborov and Rudich prove the following.

▶ **Theorem 5** (Main theorem of Razborov and Rudich as stated in [16]). *If there exists a 2^{k^ε}-hard pseudorandom generator in $\mathsf{SIZE}(k^c)$, then there is no P/poly-constructible natural property with density $1/2^{n^d}$ that is useful against the complexity class $\mathsf{SIZE}(n^e)$ whenever $e > 1 + cd/\varepsilon$.*

Broadly speaking, their proof idea is to use the *algorithm* that computes the P/poly-natural property of density ($\delta_n = \frac{1}{2^{n^d}}$) that is useful against $\mathsf{SIZE}(n^e)$ (if such a property exists) as a *distinguisher* to distinguish truth tables of functions with $\text{poly}(n)$-size circuits from truth tables of truly random functions. As this algorithm computes a natural property of density (δ_n), this means the distinguisher (which is the algorithm itself) can now distinguish a δ_n fraction of truly random functions. As δ_n here is quite large (for all n), this translates to a statistical test with a significant advantage to distinguish pseudorandom functions from truly random (n-input) functions. This in turn breaks the assumption about the existence of pseudorandom generators.

1.1.2 Technical overview

We use techniques similar to those of Razborov and Rudich to relate compression obliviousness of PARITY (and MAJORITY) with the existence of quantum-secure pseudorandom functions. We divide our contributions into two broad categories: a part that uses natural proofs, and a part that uses techniques from Fourier analysis of Boolean functions.

Part 1: quantum natural proofs

We use the term *quantum natural proofs* to mean that the proof relies on natural properties computable within a complexity class that captures quantum computation; equivalently, Υ in Definition 3 is a quantum complexity class. In our case, it is sufficient to consider Υ to be BQTIME(`linear`). Let $\mathrm{P} = (\mathrm{P}_n)_{n\in\mathbb{N}}$ be the property whose compression obliviousness, with respect to M, we want to study. Additionally, suppose that P satisfies the following two conditions.

- The property P is computable in DTIME(N); where N is the length of the input.
- For all $S \subseteq \{0,1\}^N$ of size $|S| \geq 2^N\left(1-\frac{1}{N^2}\right)$, the $\mathsf{Q}(\mathrm{P}_n|_S)$, i.e., the quantum query complexity to compute P on strings in set S, is $\Omega(N)$.

We can show that P has to be compression oblivious with respect to M unless a type of quantum secure pseudorandom functions doesn't exist.

Towards contradiction, let us assume that P is *not* compression oblivious for M. This means there exists a quantum algorithm that computes P on truth tables of functions corresponding to M in $N^{1-\alpha}$ time for some constant $\alpha > 0$. Let $\mathcal{A}$ denote this algorithm. Without loss of generality, let us assume $\mathcal{A}$ is a 1/10-bounded-error quantum algorithm. Let $\mathcal{B}$ denote the algorithm that computes P in DTIME(N) as promised earlier. Using $\mathcal{A}$ and $\mathcal{B}$, we now construct an algorithm $\mathcal{D}$ that, on all inputs z, does the following:

1. Run $\mathcal{A}$ on z for $M = O(1)$ many times and note the outputs.
2. Run $\mathcal{B}$ on z once.
3. `ACCEPT` if over M runs of $\mathcal{A}$ on input z the number of times $\mathcal{A}(z) = \mathcal{B}(z)$ is *at most* $\frac{8}{10}\cdot M$, otherwise `REJECT`. (We need the success probability of $\mathcal{D}$ to be at least 2/3 therefore choosing $M = O(1)$ suffices.)

Let z be a Boolean string of length $N = 2^n$. Let D_n denote the set of N-bit strings that $\mathcal{D}$ accepts with at least 2/3 success probability. It is easy to see that $\mathcal{D}$ rejects strings for which $\mathcal{A}$ correctly computes P (i.e., with success probability 9/10). Recall that, correctness of $\mathcal{A}$ is only promised for strings that are truth tables of functions having poly(n)-size circuits. There is no guarantee on how $\mathcal{A}$ behaves outside of this promise. But, we do know that P has a quantum query complexity of $\Omega(N)$ on all sets of size at least $2^N\left(1-\frac{1}{N^2}\right)$. As $\mathcal{A}$ is a strictly-sublinear time algorithm (hence, also a strictly-sublinear query algorithm), it can at best correctly compute P on at most $2^N\left(1-\frac{1}{N^2}\right)$ many strings outside of its promise. This, by construction, means $\mathcal{D}$ accepts at least $\frac{2^N}{N^2}$ many strings of length N and therefore D_n will contain these strings (or alternatively these functions). Now it is easy to see that $\mathcal{D}$ computes a natural property $(D_n)_{n\in\mathbb{N}}$, of density $(\delta_n = \frac{1}{2^{2n}})_{n\in\mathbb{N}}$, in $2\cdot 2^n$ quantum time. Moreover, this property $(D_n)_{n\in\mathbb{N}}$ is useful against complexity class M.

Once we have the natural property which is useful against M, we run a similar argument as done by Razborov and Rudich. As the algorithm $\mathcal{D}$ computes (D_n), it can distinguish truth tables of functions with small circuits (i.e., circuits of polylog(n)-depth and poly(n)-size or simply poly(n)-size) from truth tables of truly random functions with $\frac{1}{2^{2n}}$ advantage. But many believe that it is not possible to have quantum adversaries that are bounded by $2\cdot 2^n$ time and distinguish pseudorandom functions constructible in M with $\frac{1}{2^{2n}}$ advantage. Then it must be the case that P is compression oblivious for M.

Part 2: Fourier analysis of Boolean functions

As discussed above, our impossibility results hold for properties that are computable in DTIME(`linear`) and that have high quantum query complexity even when the computation is restricted to strings from sets of size $2^N(1-\frac{1}{N^2})$. Both PARITY and MAJORITY can be

computed in DTIME(`linear`). Moreover, using techniques from Fourier analysis, we are able to show that PARITY and MAJORITY also require $\Omega(N)$ many quantum queries on sets of size $2^N(1-\frac{1}{N^2})$. Therefore, proving Theorem 2.

1.2 Related work

- Chen *et al.* in [14] use the QSETH framework to conjecture quantum time lower bounds for variants of CNFSAT such as $\oplus$CNFSAT and #CNFSAT, etc and show lower bounds for several computational problems, such as variants of lattice problems, set-cover, hitting-set, etc., as implications.
- Arunachalam *et al.* in [7] also discuss the notion of quantum natural properties but they do so to relate learning algorithms and circuit lower bounds. As one of the results, they show that containment of the Minimum Circuit Size Problem (MCSP) in BQP implies new circuit lower bounds. This type of result was later extended by [15] to show that a few (more) quantum complexity classes can be separated if the Minimum Quantum Circuit Size Problem (MQCSP) is in BQP. Despite the overlap in topic, both of these works turn out to be complementary to the current work. We use the notion of quantum natural properties to strengthen Buhrman *et al.*'s QSETH framework by grounding assumptions in cryptography.
- Contrary to Razborov and Rudich's impossibility result, Chow [16] proves the existence of a natural proof useful to imply circuit lower bounds as a consequence of the Exponential-Time Hypothesis (ETH).

1.3 Discussion and open questions

In agreement with [11], we also believe that the notion of *compression-oblivious* properties is of independent interest. (See the blog post by Aviad Rubinstein also expressing interest [29].)

- There may be other connections with topics in meta-complexity theory (as the one we witness in this paper) or say, connections with obfuscation as suspected by Aviad Rubinstein [29] or connections with Kolmogorov complexity (as suggested by Harry Buhrman).
- Note that our results cannot be extended to show PARITY and MAJORITY are compression oblivious for AC^0 circuits. This is because AC^0-constructible pseudorandom functions are not secure against $2^{\mathrm{polylog}(n)}$-time adversaries [26]. Therefore, extending this result to PARITY or MAJORITY for AC^0_2-QSETH, i.e. CNF or DNF input, would be another step towards grounding the (necessary) assumption that such properties are compression oblivious.
- It is also worthwhile to study compression obliviousness of properties that don't meet the criteria discussed in our paper. There might be properties for which one may be able to construct natural proofs of slightly worse densities potentially implying some other complexity theoretic results as shown in [16].

On the other front, our proof technique also allows us to, in a rather black-box way, check if a particular property is likely to be compression-oblivious. One could plausibly extend our results to a larger set of functions that have a similar structure, e.g., a natural candidate would be to show an equivalent statement for symmetric functions with non-negligible mass on high-degree Fourier coefficients. It is very likely that there are in fact many properties of the type we discuss in this paper; one could use techniques similar to ones used in [3] to formally estimate the number of such properties.

1.4 Structure of the paper

In Section 2, we present useful definitions and notations. In Section 3, we formally define the QSETH framework. Our definitions omit an ambiguity from its original version presented in [11]. In Section 4, we prove our main result - the complexity-theoretic evidence for compression obliviousness of PARITY and MAJORITY. A few proofs useful for this section are moved to Appendix A. Additionally, in [13, Appendix A] in the full version of this paper, we reprove some of the earlier results about compression-oblivious properties that were given in [11]; we do this to convince the readers of the correctness of our proposed definition of the QSETH framework.

2 Preliminaries

Most of the definitions and notions presented in this section are taken from [6], and we restate them here for completeness and ease of reading.

2.1 Uniform models of computation

▶ **Definition 6** (Function and language recognition). *Let $f : \{0,1\}^* \to \{0,1\}$ be a Boolean function. The language corresponding to f is the set of strings L_f on which f evaluates to 1, i.e., $\forall x \in \{0,1\}^*$,*

$$x \in L_f \iff f(x) = 1.$$

Additionally, we say a Turing machine M decides *a language L_f if for all $x \in \{0,1\}^*$, whenever M is initialized to the start configuration on input x, the machine M halts with $f(x)$ written on its output tape.*

▶ **Definition 7** (DTIME, Definition 1.12 in [6]). *Let $T : \mathbb{N} \to \mathbb{N}$ be a function. A language L is in $DTIME(T(n))$ if there exists a Turing machine M, and there exists a constant c, such that $\forall n, \forall x \in \{0,1\}^n$, M on input x runs in $c \cdot T(n)$ steps and decides L.*

▶ **Definition 8** (BQTIME, adapting Definition 10.9 in [6]). *Let $f : \{0,1\}^* \to \{0,1\}$ and $T : \mathbb{N} \to \mathbb{N}$ be a function. We say f is computable in quantum $T(n)$-time, or (equivalently) say $L_f \in BQTIME(T(n))$, if there is a classical polynomial-time Turing machine that, for every $n \in \mathbb{N}$, on input $(1^n, 1^{T(n)})$ outputs the descriptions of a $T(n)$-size quantum circuit,[5] let us denote it by $F := (F_1, \ldots, F_{T(n)})$, such that for every $x \in \{0,1\}^n$, the output of the following process is $f(x)$ with probability at least $2/3$:*

1. *Initialize an m-qubit quantum state $|x\rangle |0^{m-n}\rangle$, where $m \leq T(n)$.*
2. *Apply one after the other $T(n)$ elementary quantum operations $F_1, \ldots, F_{T(n)}$ to the state.*[6]
3. *Measure the last qubit in the computational basis and output the measurement outcome.*

A language L_f corresponding to a Boolean function $f : \{0,1\}^* \to \{0,1\}$ is in BQP if there exists a polynomial p such that f is computable in quantum $p(n)$-time, or equivalently, $L_f \in \mathrm{BQTIME}(p(n))$.

---

[5] A few clarifications.
- A description of a T-size quantum circuit $F := (F_1, \ldots, F_T)$ is an ordered sequence of T elements, where $\forall i \in [T]$ each element $F_i = (G_i, w_i)$ specifies a quantum elementary gate $G_i \in \{\texttt{CNOT}\} \cup \{$ set of all 1-qubit operations $\}$ and the label(s) of the wire(s) w_i on which gate G_i will be applied.
- The action "applying an elementary quantum operation $F_i = (G_i, w_i)$" means applying the elementary gate G_i on wire specified by location(s) w_i.
- The size of the quantum circuit, on the other hand, is the number of elements in its description.

[6] See Footnote 5.

2.2 Non-uniform model of computation - Boolean circuits and circuit families

▶ **Definition 9** (Boolean circuits, Definition 6.1 in [6]). *For every $n \in \mathbb{N}$, an n-input, single-output Boolean circuit is a directed acyclic graph with n sources (i.e., vertices with no incoming edges but could have an unbounded number of outgoing edges) and one sink (i.e., vertex with no outgoing edges); let us denote the sink vertex by s. All non-source vertices are called gates and are labeled with one of $\wedge$, $\vee$ or $\neg$ (the logical operations AND, OR and NOT, respectively). The vertices labeled with $\wedge$ and $\vee$ have fan-in equal to 2 and those labeled with $\neg$ gate have fan-in 1. The source vertices are labeled with indices $i \in [n]$. We use $\texttt{label}(v)$ to denote the label of a vertex v. The circuit-size of C, denoted by $|\mathsf{C}|$, is the total number of vertices in it.*

If C is a Boolean circuit and $x \in \{0,1\}^n$ is some input, then the output of C on x, denoted by $\mathsf{C}(x)$, is defined in the natural way. More formally, for every vertex v of C we define a $\texttt{val}(v)$ as follows: if $\texttt{label}(v) = i$ then $\texttt{val}(v) = x_i$ and otherwise $\texttt{val}(v)$ is defined recursively by applying the logical operation of v labeled by $\texttt{label}(v)$ on the values of vertices connected to v. The output $\mathsf{C}(x)$ is the value of the sink vertex $\texttt{val}(s)$.

In any situation, if we require that the Boolean circuits are only allowed to have $\wedge$ and $\vee$ gates with fan-in 2, as opposed to these gates having unbounded fan-in, then we explicitly mention it.

▶ **Definition 10** (Circuit family and language recognition, Definition 6.2 in [6]). *Let $T : \mathbb{N} \to \mathbb{N}$ be a function. A $T(n)$-size* circuit family *is a sequence $\{C_n\}_{n\in\mathbb{N}}$ of Boolean circuits where for every n, C_n has n inputs and a single output and its circuit size $|C_n| \leq T(n)$.*

▶ **Definition 11** (SIZE). *Let $T : \mathbb{N} \to \mathbb{N}$ be a function. We say a language $L \in \mathsf{SIZE}(T)$ if there exists a $T(n)$-size circuit family $\{C_n\}_{n\in\mathbb{N}}$ such that $\forall n$, C_n consists of $\wedge$ and $\vee$ gates with fan-in 2 and $\neg$ gates of fan-in 1 and, moreover, $\forall x \in \{0,1\}^n, x \in L \iff C_n(x) = 1$.*

▶ **Definition 12** (Complexity class). *A complexity class is a set of languages.*

▶ **Definition 13** (Circuit family corresponding to a language). *Let L be a language (as in Definition 6). We say a family of circuits $\{C_n\}_{n\in\mathbb{N}}$ corresponds to language L, if $\forall n$, $\forall x \in \{0,1\}^n$, we have $x \in L \iff C_n(x) = 1$.*

▶ **Definition 14** (P/poly, Definition 6.5 in [6]). P/poly *is the class of languages that are decidable by polynomial-sized circuit families, or equivalently,* $\mathsf{P}/\mathrm{poly} = \cup_{c,d}\mathsf{SIZE}(c \cdot n^d) = \cup_{c,d}\mathsf{SIZE}_{c,d}$.

▶ **Definition 15** (AC^0_2, Adapting Definition 6.25 in [6]). *A language L is in AC^0_2 if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ and there exists a $p(n)$-size circuit family $\{C_n\}_{n\in\mathbb{N}}$ corresponding to L where $\forall n$, circuit C_n uses $\neg$, $\wedge$ and $\vee$ gates, with latter two allowed to have unbounded fan-in, and has depth at most 2.*

▶ **Definition 16** (Type-O complexity class). *We say a complexity class is of* Type-O *if it is any of the following complexity classes:*

1. *For every $d \in \mathbb{N}$ and $k \in \mathbb{N}$, we include the class $\mathsf{AC}^0_{d,k}$; we say a language $L \in \mathsf{AC}^0_{d,k}$ if there exists a family of Boolean circuits $\{C_n\}_{n\in\mathbb{N}}$ corresponding to L such that $\forall n$, C_n uses $\neg$, $\wedge$ and $\vee$ gates, with latter two allowed to have unbounded fan-in, and has depth at most d and circuit size $|C_n| \leq n^k$.*

2. *For every $d \in \mathbb{N}$ and $k \in \mathbb{N}$, we include the class $\mathsf{NC}_{d,k}$; we say a language $L \in \mathsf{NC}_{d,k}$ if there exists a family of Boolean circuits $\{C_n\}_{n\in\mathbb{N}}$ corresponding to L such that $\forall n$, C_n has depth at most $\log^d n$ and circuit size $|C_n| \leq n^k$.*
3. *For every $d \in \mathbb{N}$ and $k \in \mathbb{N}$, we include the class $\mathsf{SIZE}_{d,k}$; we say a language $L \in \mathsf{SIZE}_{d,k}$ if there exists a family of Boolean circuits $\{C_n\}_{n\in\mathbb{N}}$ corresponding to L such that $\forall n$, C_n has circuit size $|C_n| \leq d \cdot n^k$.*

2.3 Truth tables of circuits, languages and a few other useful notations

A truth table is binary string capturing the outputs of a Boolean function. If a function is defined on n binary inputs then its corresponding truth table is a Boolean string in $\{0,1\}^{2^n}$. The notion of a truth table can also be extended to capture the membership of strings in a language by fixing the length of the strings. We formally define these notions as follows.

▶ **Definition 17** (Truth tables). *For any $n \in \mathbb{N}$, let $f_n : \{0,1\}^n \to \{0,1\}$ be a Boolean function. Then the truth table of f_n, denoted by $\mathtt{tt}(f_n)$, is a 2^n-length binary string constructed in the following way:*

$$\mathtt{tt}(f_n) = \bigcirc_{a\in\{0,1\}^n} f_n(a).$$

Here $f_n(a)$ denotes the output f_n on input a. And for the concatenation, the strings $a \in \{0,1\}^n$ are taken in lexicographic order.

For a language $L \subseteq \{0,1\}^*$ the notion of truth table is as follows: for any $n \in \mathbb{N}$, the truth table of L at n, denoted by $\mathtt{tt}(L,n)$, is a 2^n-length binary vector that can be obtained by concatenating the outputs $b_a := [a \in L]$ for all $a \in \{0,1\}^n$ taken in lexicographic order.

Other useful notations

Let n be an integer. Let $f_n : \{0,1\}^n \to \{0,1\}$ be a Boolean function on n input variables.

- We use F_n to denote the set of *all* Boolean functions defined on n input variables; We represent $F_n = \{\mathtt{tt}(f_n) \mid f_n \text{ is a Boolean function on } n \text{ variables}\}$ as the set of truth tables of all functions $f_n : \{0,1\}^n \to \{0,1\}$, implying $F_n = \{0,1\}^{2^n}$ and $|F_n| = 2^{2^n}$. We also use $N = 2^n$.
- We overload the notation of $\mathtt{tt}(\cdot)$ on sets of Boolean functions to denote the following: if S is a set of Boolean functions on n variables then $\mathtt{tt}(S) = \{\mathtt{tt}(f_n) \mid f_n \in S\}$.
- Let Λ be a complexity class. Let $L \in \Lambda$. Furthermore, let $f_n : 0,1^n \to \{0,1\}$ be a Boolean function such that $\mathtt{tt}(f_n) = \mathtt{tt}(L,n)$ then we say f_n *is a sub-element of* Λ.
- We sometimes also overload the notation of $\mathtt{tt}(\cdot)$ on complexity classes to denote the following: if Λ is a complexity class then $\mathtt{tt}(\Lambda,n) = \{\mathtt{tt}(f_n) \mid f_n \text{ sub-element in } \Lambda\}$ or equivalently $\mathtt{tt}(\Lambda,n) = \{\mathtt{tt}(L,n) \mid L \in \Lambda\}$. Furthermore, we use $\mathtt{tt}(\Lambda) = \bigcup_{n\in\mathbb{N}} \mathtt{tt}(\Lambda,n)$.

3 The QSETH framework

3.1 Definitions

▶ **Definition 18** (Property). *A Boolean property (or just property) is a sequence $P := (P_n)_{n\in\mathbb{N}}$ where each P_n is a set of Boolean functions defined on n variables.*

Alternatively, one can view P_n as a set of 2^n-length Boolean strings that are truth tables of functions $f_n \in \mathrm{P}_n$, or equivalently view it as a function $\mathrm{P}_n : \{0,1\}^{2^n} \to \{0,1\}$ such that $\mathrm{P}_n(\mathtt{tt}(f_n)) = 1$ if and only if $f_n \in \mathrm{P}_n$; see Definition 17 for the definition of a truth table. Also note that it is possible for a property P to be *partial*. In such a case, for every n, there could be some functions f_n for which we don't care whether or not they belong to P_n.[7]

[7] For example, both the properties $\mathrm{PP}_{\mathrm{edit}}$ and $\mathrm{PP}_{\mathrm{lcs}}$ considered in [11] are partial.

Quantum query complexity of P

The (bounded-error) quantum query complexity is defined only in a non-uniform setting; therefore, we define this notion for P_n for every $n \in \mathbb{N}$. A quantum query algorithm $\mathcal{B}$ for $\mathrm{P}_n : \{0,1\}^{2^n} \to \{0,1\}$, on an input $x \in \{0,1\}^{2^n}$ begins in a fixed initial state $|\psi_0\rangle$, applies a sequence of unitaries $U_0, O_x, U_1, O_x, \ldots, U_T$, and performs a measurement whose outcome is denoted by z. Here, the initial state $|\psi_0\rangle$ and the unitaries $U_0, U_1, \ldots, U_T$ are independent of the input x. The unitary O_x represents the "query" operation, and maps $|i\rangle\,|b\rangle$ to $|i\rangle\,|b + x_i \bmod 2\rangle$ for all $i \in [2^n] - 1$. We say that $\mathcal{B}$ is a 1/3-bounded-error algorithm computing P_n if for all x in the domain of P_n, the success probability of outputting $z = \mathrm{P}_n(x)$ is at least 2/3. Let $\mathrm{cost}(\mathcal{B})$ denote the number of queries $\mathcal{B}$ makes to O_x throughout the algorithm. The 1/3-bounded-error quantum query complexity of P_n, denoted by $\mathsf{Q}_{1/3}(\mathrm{P}_n)$, is defined as $\mathsf{Q}_{1/3}(\mathrm{P}_n) = \min\{\mathrm{cost}(\mathcal{B}) : \mathcal{B} \text{ computes } \mathrm{P}_n \text{ with error probability } \leq 1/3\}$.[8]

Algorithms computing a property P in black-/white-box setting

We say a (bounded-error quantum) algorithm $\mathcal{A}$ computes a property P (as in Definition 18) in the *black-box* setting if for all $n \in \mathbb{N}$, for all f_n given its truth table as input, i.e., a 2^n-length Boolean string, $\mathcal{A}$ makes (quantum) queries to this input and (with success probability at least 2/3) determines whether or not $f_n \in \mathrm{P}_n$. An algorithm computing P in the *white-box* setting gets a circuit description of f_n, which is possibly a more succinct description of f_n's truth table, and the algorithm has to determine whether or not $f_n \in \mathrm{P}_n$. We emphasize that, in both the black-/white-box setting, we care about the *time* that algorithm $\mathcal{A}$ takes when deciding whether or not $f_n \in \mathrm{P}_n$.

We now present the definition of compression-oblivious properties. In simple terms, these are the properties whose quantum query complexity is a lower bound for the time complexity to compute the property even for inputs that are "compressible" as a truth table of small formulas (or circuits). Formally,

▶ **Definition 19** ($\Gamma_{d,k}$-compression-oblivious properties). *Let $d, k \in \mathbb{N}$. Let $\Gamma_{d,k}$ denote a* Type-O *complexity class as stated in Definition 16 parameterized by d and k. We say a property P is* $\Gamma_{d,k}$-compression-oblivious*, denoted by $P \in \mathcal{CO}(\Gamma_{d,k})$, if for every constant $\delta > 0$, for every quantum algorithm $\mathcal{A}$ that computes P in the **black-box** setting, $\forall n' \in \mathbb{N}, \exists n \geq n'$ and $\exists$ a set $L = \{L^1, L^2, \ldots\} \subseteq \Gamma_{d,k}$ of "hard languages", such that $\forall$ circuit families $\{C^1_{n''}\}_{n'' \in \mathbb{N}}$ corresponding to L^1, $\forall$ circuit families $\{C^2_{n''}\}_{n'' \in \mathbb{N}}$ corresponding to L^2, ..., the algorithm $\mathcal{A}$ uses at least $\mathsf{Q}_{1/3}(P_n)^{1-\delta}$ quantum time on at least one of the inputs in $\{C^i_n\}_{i \in [|L|]}$.*[9]

▶ **Definition 20** (compression-oblivious properties w.r.t. AC^0_2, NC, P/poly). *We say a property $P : \{0,1\}^* \to \{0,1\}$ is*

1. $P \in \mathcal{CO}(\mathsf{AC}^0_2)$ *if $d = 2$ and $\exists k \in \mathbb{N}$ such that $P \in \mathcal{CO}(\mathsf{AC}^0_{d,k})$,*
2. $P \in \mathcal{CO}(\mathsf{AC}^0)$ *if $\exists d \in \mathbb{N}, \exists k \in \mathbb{N}$ such that $P \in \mathcal{CO}(\mathsf{AC}^0_{d,k})$,*
3. $P \in \mathcal{CO}(\mathsf{NC})$ *if $\exists d \in \mathbb{N}, \exists k \in \mathbb{N}$ such that $P \in \mathcal{CO}(\mathsf{NC}_{d,k})$, and*
4. $P \in \mathcal{CO}(\mathrm{P/poly})$ *if $\exists d \in \mathbb{N}, \exists k \in \mathbb{N}$ such that $P \in \mathcal{CO}(\mathsf{SIZE}_{d,k})$.*

[8] Note that if $\textsc{parity} := (\textsc{parity}_n)_{n \in \mathbb{N}}$ is the property we are considering, under our definition of $\textsc{parity}_n$ the value of $\mathsf{Q}(\textsc{parity}_n)$ is equal to $\Omega(2^n)$ and not $\Omega(n)$. For the property $\mathrm{OR} := (\mathrm{OR}_n)_{n \in \mathbb{N}}$ then $Q(\mathrm{OR}_n) = \Omega(2^{\frac{n}{2}})$.

[9] While the idea of compression-oblivious properties can be easily expressed informally, describing the notion formally requires discussing intricate details to bridge the connections between uniform (e.g., algorithm $\mathcal{A}$ computing P) and non-uniform models of computations (e.g., query complexity $\mathsf{Q}(\mathrm{P}_n)$).

The original definition of compression-oblivious properties, as presented in Section 2.2 of [11], does not clearly enforce a unique ordering of the quantifiers, which can potentially lead to undesirable consequences. Hence, we make the effort of presenting a more formal definition of Γ-compression-oblivious properties in Definitions 19 and 20 (with the approval of the authors [11]).

Having formally defined Γ-compression-oblivious properties for all $\Gamma \in \{\mathsf{AC}^0_2, \mathsf{AC}^0, \mathsf{NC}, \mathsf{P/poly}\}$ we now state the Γ-QSETH conjecture. Earlier in this section, we discussed the difference between computing a Boolean property in a black-box setting versus computing the same property in a white-box setting. The compression-oblivious properties are those properties for which its corresponding quantum query complexity lower bounds the time complexity in the *black-box* setting. For such properties, Buhrman *et al.* conjectured that the time complexity of computing these properties in the *white-box* setting is as bad as it is in the black-box setting. This assumes that the succinct description of the Boolean functions doesn't help in computing the properties more efficiently. (This is similar to the *Black-Box Hypothesis (BBH)* stated in [22].) More formally,

▶ **Conjecture 21** (Γ-QSETH, a more formal presentation of Conjecture 5 in [11])**.** *Let* $\Gamma \in \{\mathsf{AC}^0_2, \mathsf{AC}^0, \mathsf{NC}, \mathsf{P/poly}\}$ *be a complexity class (as in Definition 12). Let* P *be a* Γ-compression-oblivious *property. The* Γ*-QSETH conjecture states that* $\exists d, \exists k \in \mathbb{N}$*, such that for every constant* $\delta > 0$*, for every quantum algorithm* $\mathcal{A}$ *that computes* P *in the* ***white-box*** *setting,* $\forall n' \in \mathbb{N}, \exists n \geq n'$*,* $\exists$ *a set* $L = \{L^1, L^2, \ldots\} \subseteq \Gamma_{d,k}$*,* $\forall$ *circuit families* $\{C^1_{n''}\}_{n'' \in \mathbb{N}}$ *corresponding to* L^1*,* $\forall$ *circuit families* $\{C^2_{n''}\}_{n'' \in \mathbb{N}}$ *corresponding to* L^2*, ..., the algorithm* $\mathcal{A}$ *uses at least* $\mathsf{Q}_{1/3}(P_n)^{1-\delta}$ *quantum time on at least one of the inputs in* $\{C^i_n\}_{i \in [|L|]}$*. Note that* $\Gamma_{d,k}$ *is a* Type-O *complexity class corresponding to* Γ*.*

3.2 Relevance

Note that a basic version of the QSETH conjecture and its implications has already been discussed in complementary works [1, 11], and other variants of QSETH are discussed more recently in [14] and in Chen's PhD thesis [12]. For example,

▶ **Conjecture 22** (Basic-QSETH, Conjecture 8.3 in [12])**.** *For each constant* $\delta > 0$*, there exists* $c > 0$ *such that there is no bounded-error quantum algorithm that solves CNFSAT (even restricted to formulas with at most* cn^2 *many clauses) in* $O(2^{\frac{n(1-\delta)}{2}})$ *time.*

▶ **Conjecture 23** (⊕QSETH, Conjecture 8.9 in [12])**.** *For each constant* $\delta > 0$*, there exists* $c > 0$ *such that there is no bounded-error quantum algorithm that solves* $\oplus$*CNFSAT (even restricted to formulas with at most* cn^2 *many clauses) in* $O(2^{n(1-\delta)})$ *time.*

▶ **Conjecture 24** (#QSETH, Conjecture 8.8 in [12])**.** *For each constant* $\delta > 0$*, there exists* $c > 0$ *such that there is no bounded-error quantum algorithm that solves #CNFSAT (even restricted to formulas with at most* cn^2 *many clauses) in* $O(2^{n(1-\delta)})$ *time.*

We are able to get these conjectures as a corollary to our QSETH framework. More precisely,

▶ **Corollary 25.** *The* AC^0_2*-QSETH implies Conjecture 22. Additionally, if we assume that* $\textsc{parity} \in \mathcal{CO}(\mathsf{AC}^0_2)$ *then* AC^0_2*-QSETH implies Conjectures 23 and 24.*

Proof. The proof of $\mathrm{OR} \in \mathcal{CO}(\mathsf{AC}^0_2)$ was first given in [11] (we also present the proof in the full version of this paper). Thereafter, setting $d = 2$ and $k = 3$ for AC^0_2-QSETH conjecture implies Conjectures 22–24. ◀

Therefore, all the QSETH-based lower bounds presented in Chen's PhD thesis [12] immediately follow from our QSETH framework.

4 When certain properties are not P/poly-compression-oblivious

In this section, we show that it is reasonable to conjecture that there exist $d', k' \in \mathbb{N}$ and $d'', k'' \in \mathbb{N}$ for which properties such as PARITY and MAJORITY are $\mathsf{SIZE}_{d',k'}$-compression-oblivious and $\mathsf{SIZE}_{d'',k''}$-compression-oblivious, respectively. We do so by relating this conjecture to topics in circuit complexity and pseudorandomness. In fact, we are able to show similar consequences for properties that satisfy certain constraints. PARITY and MAJORITY happen to be such properties.

Following the notation about natural proofs from [28] and [16], we now define the following.

▶ **Definition 26** (Υ-natural property with density δ). *Let Υ be a complexity class (as in Definition 12) and let $\delta := (\delta_n)_{n\in\mathbb{N}}$ denote a sequence of positive real numbers. A property $D := (D_n)_{n\in\mathbb{N}}$ (as in Definition 18) is Υ-natural with density $\delta := (\delta_n)_{n\in\mathbb{N}}$ if the following statements hold:*

1. *largeness: $\exists n', \forall n > n'$, $|D_n| \geq 2^{2^n} \cdot \delta_n$, and*
2. *constructivity: the problem of determining whether a Boolean function $f_n \in D_n$, when given as input the truth table of f_n on n variables, is decidable in Υ, which in more formal terms translates to saying $L_D \in \Upsilon$ where $L_D := \bigcup_{n\in\mathbb{N}}\{\mathtt{tt}(f_n) \mid f_n \in D_n\}$.*

▶ **Definition 27** (Quasi-useful property, Definition 5 in [16]). [10] *Let Λ denotes a complexity class. A property $D := (D_n)$, or equivalently the corresponding language $L_D := \bigcup_{n\in\mathbb{N}}\{\mathtt{tt}(g_n) \mid g_n \in D_n\}$, is quasi-useful against complexity class Λ if $\forall(f_n)$, we have $(f_n) \notin \Lambda$ whenever (f_n) is sequence of Boolean functions satisfying $f_n \in D_n$ for all sufficiently large n.*

Our goal is to realize the chain of arguments as illustrated in Figure 1.

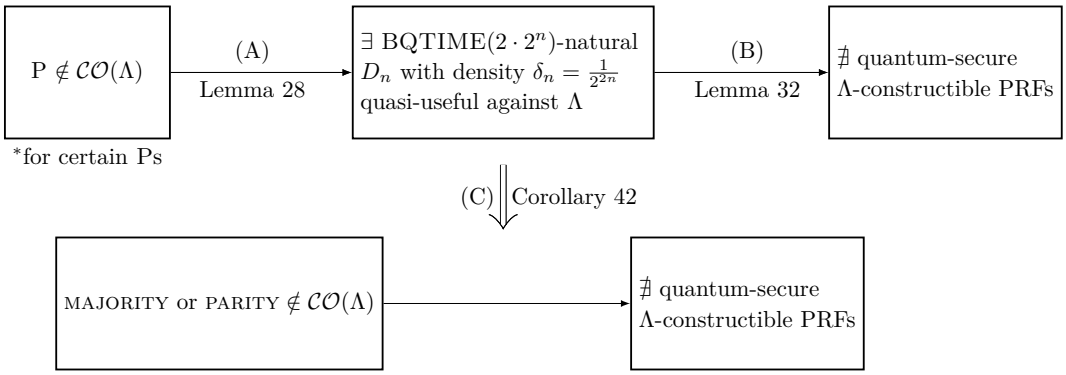


Figure 1 Chain of arguments presented in this section. We discuss the implications when a certain class of properties P are not Λ-compression-oblivious in links (A) and (B). In link (C) we show that MAJORITY and PARITY belong to this class of properties.

[10] Note that Razborov and Rudich use the notion of *useful* property which is defined for infinitely many n. However, for our results the notion of *quasi-useful* property (defined for all sufficiently large n) makes more sense so we use this instead.

Realising link (A)

The proof idea is as follows. Let $\Lambda = \mathsf{SIZE}_{d,k}$. Our choice of d, k will be clear later. If PARITY $\notin \mathcal{CO}(\Lambda)$ (or say MAJORITY $\notin \mathcal{CO}(\Lambda)$ may be for another choice of d, k) then (from the definition of compression-oblivious properties) there exists a quantum algorithm that computes the PARITY (or MAJORITY) of truth tables of circuits C that are sub-elements of Λ in time sublinear in the size of the input $\mathtt{tt}(C)$. Using this algorithm and the linear time deterministic algorithm to compute PARITY (or MAJORITY) on all strings of length $|\mathtt{tt}(C)|$, we will construct a BQTIME($\mathtt{linear}$)-natural property that is quasi-useful against complexity class Λ. Interestingly enough, the idea also generalizes to other simple properties that satisfy certain requirements; we formalize that in the statement of Lemma 28.

▶ **Lemma 28.** *Let $d, k \in \mathbb{N}$ and $\Lambda = \mathsf{SIZE}_{d,k}$. If a property $P : \{0,1\}^* \to \{0,1\}$, or alternatively denoted as $(P_n)_{n\in\mathbb{N}}$ (as in Definition 18), satisfies the following conditions:*

1. *$P \notin \mathcal{CO}(\Lambda)$,*
2. *P on inputs of length N is computable in $DTIME(N)$, which more formally means $L_P := \bigcup_{n\in\mathbb{N}}\{\mathtt{tt}(f_n) \mid f_n \in P_n\} \in DTIME(N)$, and*
3. *$\forall S \subseteq \{0,1\}^N$ of size $|S| \geq 2^N(1 - \frac{1}{N^2})$, any 1/3-bounded-error quantum query algorithm that computes $P_{\log N}$ on each $x \in S$ requires $\Omega(N)$ queries,*

then there exists a $BQTIME(2 \cdot N)$-natural property (D_n) with density $\delta_n = \frac{1}{N^2}$ and this natural property is quasi-useful against complexity class Λ.

Proof. Let $d, k \in \mathbb{N}$ and let $\Lambda = \mathsf{SIZE}_{d,k}$. From Definition 19 we say, $\mathrm{P} \notin \mathcal{CO}(\Lambda)$ if and only if $\exists \delta > 0$, $\exists$ algorithm $\mathcal{A}$ computing property P such that $\exists n' \in \mathbb{N}$ and $\forall n \geq n'$, $\forall L = \{L^1, L^2, \ldots\} \subseteq \Lambda$, $\mathcal{A}$ uses at most $Q_{1/3}(\mathrm{P}_n)^{1-\delta}$ quantum time on all inputs in $\{\mathtt{tt}(L^i, n)\}_{i\in[|L|]}$.

Then, let $n \geq n'$, let $N = 2^n$ and let L be the largest subset of Λ, i.e., the set Λ itself, $\mathcal{A}$ computes the property in at most $Q_{1/3}(\mathrm{P}_n)^{1-\delta}$ time for all binary strings in $\mathtt{tt}(\Lambda) = \{\mathtt{tt}(L^i, n)\}_{i\in[|L|]}$. Furthermore, we have $|\mathtt{tt}(\Lambda)| \leq 2^{d' \cdot n^k}$ for some $d' > d$; this is because all the languages $L^i \in \Lambda$ which means for every L^i there must exist a circuit family $\{C^i_n\}_{n\in\mathbb{N}}$ with $|C_n| \leq d \cdot n^k$ and the number of such circuits is at most $2^{d' \cdot n^k}$ which means the number of *unique* 2^n-length truth tables that Λ can hold is at most $2^{d' \cdot n^k}$. W.l.o.g., let us assume that $\mathcal{A}$ computes P with success probability at least 9/10 (as we can boost the probability of success by running the 1/3-bounded-error algorithm $\mathcal{A}$ and then take majority voting). For all $n > n'$, for all $x \in \mathtt{tt}(\Lambda, n)$, the algorithm $\mathcal{A}$ computes $\mathrm{P}(x)$ in $\mathsf{Q}_{1/10}(\mathrm{P}_{\log|x|})^{1-\delta} \leq |x|^{1-\delta}$ time for some constant $\delta > 0$.

In addition, let $\mathcal{B}$ denote a deterministic (quantum or classical) algorithm that computes P on all $y \in \{0,1\}^*$ in $\Theta(|y|)$ time – upper and lower bound follow from Item 2 and Item 3 of the theorem statement, respectively. Using $\mathcal{A}$ and $\mathcal{B}$ we now construct Algorithm $\mathcal{D}$ that on an input $z \in \{0,1\}^N$ does the following:

1. Run $\mathcal{A}$ on z for $M = O(1)$ times and note the outputs.
2. Run $\mathcal{B}$ on z once.
3. `Accept` if over M runs of $\mathcal{A}$ on input z the number of times $\mathcal{A}(z) = \mathcal{B}(z)$ is *at most* $\frac{8}{10} \cdot M$, otherwise `reject`. We need the success probability of $\mathcal{D}$ to be at least 2/3 therefore choosing $M = O(1)$ suffices.

Notice that Algorithm $\mathcal{D}$ (with high probability) computes a property that is quasi-useful (Definition 27) against complexity class Λ. This is true because $\mathcal{B}(z) = \mathrm{P}(z)$ for all z and whenever $z \in \mathtt{tt}(\Lambda)$ the $\Pr[\mathcal{A}(z) = \mathrm{P}(z)] \geq 0.9$, which means the outputs $\mathcal{A}(z)$ and $\mathcal{B}(z)$ are equal more than 9/10 of the M times. Therefore, if $z \in \mathtt{tt}(\Lambda)$ then $\mathcal{D}$ rejects z.

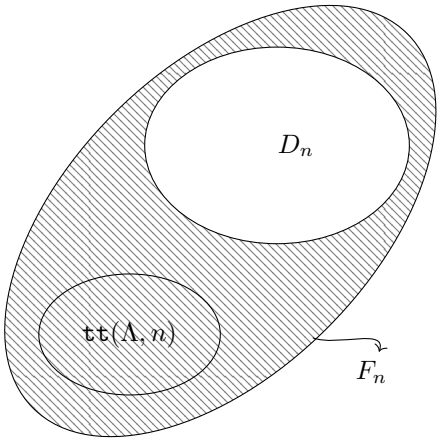


Figure 2 A pictorial representation of the fraction of 2^n-length strings that the algorithm $\mathcal{D}$ accepts. Recall that F_n denotes the full set $\{0,1\}^{2^n}$; hence, $|F_n| = 2^{2^n}$. The set $\mathtt{tt}(\Lambda, n)$ denotes the set of strings $\{\mathtt{tt}(L, n) \mid L \in \Lambda\}$. Algorithm $\mathcal{A}$ is guaranteed to accept all strings from $\mathtt{tt}(\Lambda, n)$ with success probability at least 9/10 and no guarantee whatsoever on strings in $F_n \setminus \mathtt{tt}(\Lambda, n)$. The set D_n denotes the set of 2^n-length strings that Algorithm $\mathcal{D}$ accepts.

What is left is to argue that Algorithm $\mathcal{D}$ computes a BQTIME$(2 \cdot N)$-natural property that satisfies both the *largeness* and *contructivity* conditions as mentioned in Definition 26. Let $z \in F_n$ and let $D_n = \{z \mid |z| = 2^n$ and Algorithm $\mathcal{D}$ accepts $z\}$.

- Algorithm $\mathcal{D}$ on any z of size $N = 2^n$ can in at most $2 \cdot N$ quantum time decide whether or not $z \in D_n$; hence satisfying the *constructivity* condition.
- We argue about the largeness condition of $|D_n| \geq 2^N \cdot \delta_n$ for $\delta_n = \frac{1}{N^2}$ in the following way. Consider the set F_n, i.e., the set of truth tables of all Boolean functions defined on n input variables. For a string $z \in F_n$, the algorithm $\mathcal{D}$ rejects z only if $\Pr[\mathcal{A}(z) = \mathrm{P}(z)] > 8/10$. In other words, $\mathcal{D}$ rejects z whenever $z \in F_n \setminus D_n$; see Figure 2. The runtime of Algorithm $\mathcal{A}$ on N-length inputs is $O(N^{1-\beta})$ time (for a constant $\beta > 0$), and we know from Item 3 in the statement of the theorem that $\mathcal{A}$ cannot compute P correctly (with success probability greater than 2/3) on sets bigger than $2^N(1 - \frac{1}{N^2})$. Hence, $|F_n \setminus D_n| \leq 2^N \cdot (1 - \frac{1}{N^2})$, which means $|D_n| \geq \frac{2^N}{N^2}$.

This concludes the proof of *Lemma* 28. ◀

Realising link (B)

We will now show the cryptographic implications when properties like PARITY and MAJORITY are not compression-oblivious. In link (A) we showed that, if either of these properties is not compression-oblivious with respect to complexity class $\Lambda = \mathsf{SIZE}_{d,k}$ for some constants $d, k \in \mathbb{N}$, then we can construct a BQTIME$(2 \cdot 2^n)$-natural property useful against Λ. Any quantum algorithm computing this natural property can now be used to distinguish pseudorandom functions constructible in $\Lambda = \mathsf{SIZE}_{d,k}$ from truly random functions. However, such linear-time (linear in the length of the truth table of the functions) quantum distinguishers are not believed to exist. Therefore, it is plausible that a quantum algorithm computing this natural property in the specified time does not exist.

Let us recall the standard definition of pseudorandom functions [18].

▶ **Definition 29** (Keyspace). *Let $K : \mathbb{N} \to \mathbb{N}$ be a function. A key space $\mathcal{K}$ generated by the function K is a set of sets defined as $\mathcal{K} = \{\{0,1\}^{K(n)}\}_{n\in\mathbb{N}}$ and we use $\mathcal{K}(n)$ to denote the set $\{0,1\}^{K(n)}$.*

Note that, for our results, the function K is always chosen to be a polynomial.

▶ **Definition 30** (Pseudorandom functions (PRFs)).
A set $F = \{ f_k : \{0,1\}^n \to \{0,1\} \}_{n\in\mathbb{N},k\in\mathcal{K}(n),}$ is called a family of pseudorandom functions with key space $\mathcal{K}$ generated by function K if for all probabilistic polynomial time (p.p.t.) adversary A, there is a negligible function ε such that for all $n \in \mathbb{N}$,

$$|\Pr_k[A^{f_k(\cdot)}(1^n) = 1] - \Pr_{\mathsf{TR}}[A^{\mathsf{TR}(\cdot)}(1^n) = 1]| < \eta(n),$$

where the probability is also over the random string used by A, TR denotes a truly random function from $\{0,1\}^n$ to $\{0,1\}$.

*Sometimes we may strengthen or weaken the definition by letting the adversary's running time be $T(n)$-bounded, and letting the distinguishing probability to be less than some specific function $\eta(n)$. In this case, we call it a (K,T,η)-*PRF*. Additionally, if the family F, which can be interpreted as a families of circuits, corresponds to languages in Λ, then we call it as Λ-constructible (K,T,η)-*PRF*s.*

Pseudorandom functions can also be defined against quantum adversaries making (classical or) quantum queries; see [32] for example. Formally stated as follows.

▶ **Definition 31** (Quantum-secure PRFs). *A set $F = \{ f_k : \{0,1\}^n \to \{0,1\} \}_{k\in\mathcal{K}(n),n\in\mathbb{N}}$ is called a family of quantum-secure pseudorandom functions with key space $\mathcal{K}$ generated by a function K if for all polynomial time (polynomial in the length of its input) quantum adversary A making quantum queries, there is a negligible function ε such that for all $n \in \mathbb{N}$,*

$$|\Pr_k[A^{f_k(\cdot)}(1^n) = 1] - \Pr_{\mathsf{TR}}[A^{\mathsf{TR}(\cdot)}(1^n) = 1]| < \eta(n),$$

where the probability is also over the random string used by A, TR denotes a truly random function from $\{0,1\}^n$ to $\{0,1\}$.

Sometimes (just as we do for PRF*s) we may strengthen or weaken the definition by letting the adversary's running time be $T(n)$-bounded, and letting the distinguishing probability governed by some other function $\eta(n)$. In this case, we denote them by (K,T,η)-quantum-secure-*PRF*s. Additionally, if the family F, which can be interpreted as a families of circuits, correspond to languages in Λ, then we call it as Λ-constructible (K,T,η)-quantum-secure-*PRF*s.*

Using this notion of quantum-secure pseudorandom functions, we are able to prove the following.

▶ **Lemma 32.** *Let $d,k \in \mathbb{N}$. If there exists a $BQTIME(2\cdot 2^n)$-natural property of density $\delta_n \geq \frac{1}{2^{2n}}$ that is quasi-useful against the complexity class $\Lambda = \mathsf{SIZE}_{d,k}$ then there doesn't exist any Λ-constructible $(d\cdot n^k, 2\cdot 2^n, \frac{1}{2^{2n}})$-quantum-secure-*PRF*; the size of the truth table of function f_n is 2^n.*

Proof. Let $N = 2^n$ be the length of the truth table of functions on n variables. Let P be a BQTIME$(2\cdot N)$-natural property of density $\delta_n \geq \frac{1}{2^{2n}}$ that is quasi-useful against the complexity class $\mathsf{SIZE}_{d,k}$. Let $\mathcal{C}$ be the algorithm that computes this property P. Because $\mathcal{C}$ computes P, for any (sufficiently large) value of n the following statements hold.

- Algorithm $\mathcal{C}$ rejects z whenever z is a string from $\mathtt{tt}(\mathsf{SIZE}_{d,k}, n)$; this is because P is quasi-useful against $\mathsf{SIZE}_{d,k}$.
- There are at least $\frac{1}{2^{2n}} \cdot 2^{2^n}$ number of functions in F_n that $\mathcal{C}$ accepts; this is because the P has density $\delta_n \geq \frac{1}{2^{2n}}$.
- Furthermore, $\mathcal{C}$ runs in $\mathrm{BQTIME}(2 \cdot 2^n)$.

Combining these observations we can see that

$$|\Pr[\mathcal{C}(x) = 1] - \Pr[\mathcal{C}(y) = 1]| \geq \frac{1}{2^{2n}}$$

when x is chosen at random from $\mathtt{tt}(\mathsf{SIZE}_{d,k}, n)$ and y is chosen at random from F_n. This means $\mathcal{C}$, despite being an algorithm running in $\mathrm{BQTIME}(2 \cdot N)$ time, can distinguish $\mathsf{SIZE}_{d,k}$-constructible pseudorandom functions from the set of all random functions. Hence, showing that $\mathsf{SIZE}_{d,k}$-constructible $(d \cdot n^k, 2 \cdot 2^n, \frac{1}{2^{2n}})$-quantum-secure-PRFs do not exist. ◀

We could also state our assumption about the existence of quantum-secure PRFs as a conjecture.

▶ **Conjecture 33.** *There exists $d, k \in \mathbb{N}$ for which $\mathsf{SIZE}_{d,k}$-constructible $(d \cdot n^k, 2 \cdot 2^n, \frac{1}{2^{2n}})$-quantum-secure-PRFs exist.*

Using Lemma 28 and Lemma 32 we can now claim the following statement.

▶ **Theorem 34.** *Let $P : \{0,1\}^* \to \{0,1\}$ be the Boolean property satisfying Items 2 and 3 stated in Lemma 28. Then property $P \in \mathcal{CO}(\mathsf{P/poly})$ unless Conjecture 33 is false.*

Proof. If $\mathrm{P} \notin \mathcal{CO}(\mathsf{P/poly})$ then $\forall d, \forall k \in \mathbb{N}$ we have $\mathrm{P} \notin \mathcal{CO}(\mathsf{SIZE}_{d,k})$; see Definition 20. For every value of d, k we will then be able to construct a $\mathrm{BQTIME}(2 \cdot 2^n)$-natural property that will be quasi-useful against $\mathsf{SIZE}_{d,k}$. This means for all values of d, k we will no longer have $\mathsf{SIZE}_{d,k}$-constructible $(d \cdot n^k, 2 \cdot 2^n, \frac{1}{2^{2n}})$-quantum-secure-PRFs. Hence, falsifying Conjecture 33. ◀

▶ Remark 35. Although not explicitly used in this paper, we would also like to mention that under the assumption that the learning with errors (LWE) conjecture holds against quantum algorithms, there exist quantum-secure PRFs evaluatable by NC^2 circuits (NC^1 if we assume the quantum hardness of RingLWE) [9]. This suggests that under the quantum hardness of LWE (resp. RingLWE), PARITY and MAJORITY are compression obliviousness with respect to the complexity class NC^2 (resp. NC^1).

▶ Remark 36 (Strengthening Conjecture 33 with OWFs). First, recall the definition of one-way functions.

▶ Definition 37 (One-way functions (OWFs)). *A function $g : \{0,1\}^* \to \{0,1\}^*$ is called a (T, η)-one-way function if there is a polynomial p such that for all adversary A running in time T, for all $n \in \mathbb{N}$,*

$$\Pr_{x \leftarrow \{0,1\}^{p(n)}}[A(1^n, g(x)) = x' \text{ such that } g(x') = g(x)] < \eta(n).$$

For example, when $T = 2^n$, we can set the length of the domain and range to be $p(n) = n^2$, therefore a $T = 2^n$ adversary will not trivially break such an OWF.

It is known [20, 18] that (K, T, η)-PRF exists if $(\mathrm{poly}(T), \mathrm{poly}(\eta))$-one-way function exists. Furthermore, the same result holds against quantum adversaries even if the adversary for the PRF is allowed to make superposition queries [32]. Therefore, we can say that

$(2^n, 2^{-\omega(n)})-$PRF against quantum adversaries exists as long as $(\text{poly}(2^n), \text{poly}(2^{-\omega(n)}))$-one-way function against quantum adversaries exists. Therefore, the following conjecture implies Conjecture 33.

▶ Conjecture 38. *Quantum secure* $(\text{poly}(2^n), \text{poly}(2^{-\omega(n)}))$*-one-way functions exist.*

Realising implication (C)

Earlier in this section we discussed that the arguments presented work for properties that satisfy Items 1–3 in the statement of Lemma 28. For properties like PARITY or MAJORITY it is trivial to see that they admit a linear time (quantum and classical) deterministic algorithm. However, it is not immediately clear if the query complexity of computing these properties is "high" on sets $S \subset F_n$ that are not the full set F_n but still are relatively "large". Using techniques from Fourier analysis of Boolean properties, we can show that this is indeed the case with PARITY and MAJORITY. In fact, we are able to say something more. More precisely, we show that the following statement holds.

▶ **Lemma 39.** *Let N be an integer and let $S \in \{0,1\}^N$ be a subset of size $s \geq 2^N\left(1 - \frac{1}{N^2}\right)$. Let $P : \{0,1\}^N \to \{-1,1\}$ be a Boolean property with $Q(P) = \Omega(N)$ and the N-degree Fourier coefficient $|\hat{P}([N])| \geq \sqrt{\frac{1}{2N}}$.*[11] *Then every quantum algorithm that, with error probability $\varepsilon \leq \frac{1}{2\cdot N^2}$, computes P on each $x \in S$ uses $\Omega(N)$ queries.*

Proof. Suppose there exists a T-query algorithm, let us denote by $\mathcal{A}_S$, to compute property P on all $x \in S$ with worst-case error probability $\varepsilon \leq \frac{1}{2\cdot N^2}$. W.l.o.g, we can assume that the output of $\mathcal{A}_S$ is in $\{-1,1\}$. Let $R(x)$ denote the probability that $\mathcal{A}_S$ outputs -1; then we can immediately see that $\forall x \in S$,

$$(1 - 2R(x)) \cdot \mathrm{P}(x) \geq (1 - 2\varepsilon).^{12} \tag{1}$$

We will now estimate the degree of the polynomial $R(x)$. Observe that $\deg(R(x)) = \deg(1 - 2R(x))$. Let $Q(x) = (1 - 2R(x))$ and let $\varepsilon' = (1 - 2\varepsilon)$. Let the Fourier expansions of $Q(x) = \sum_{B\subseteq[N]} \hat{\mathrm{Q}}(B)\chi_B(x)$ and $\mathrm{P}(x) = \sum_{B\subseteq[N]} \hat{\mathrm{P}}(B)\chi_B(x)$, respectively.

Towards a contradiction, suppose that $\deg(Q(x)) < N$ then $\hat{\mathrm{Q}}([N]) = 0$, which implies

$$\begin{aligned}\left(\sum_{B\in 2^{[N]}} \hat{\mathrm{Q}}(B)\cdot\hat{\mathrm{P}}(B)\right)^2 &\overset{(1)}{=} \left(\sum_{B\in 2^{[N]}\setminus[N]} \hat{\mathrm{Q}}(B)\cdot\hat{\mathrm{P}}(B)\right)^2 \\ &\overset{(2)}{\leq} \left(\sum_{B\in 2^{[N]}\setminus[N]} \hat{\mathrm{Q}}(B)^2\right)\cdot\left(\sum_{B\in 2^{[N]}\setminus[N]} \hat{\mathrm{P}}(B)^2\right) \\ &\overset{(3)}{\leq} 1 - \frac{1}{2N};\end{aligned}$$

[11] Note that we use these constants so that Lemma 39 can be almost directly used for PARITY and MAJORITY. However, one can modify these constants to accommodate other properties as well.

[12] Since $R(x)$ is the probability that $\mathcal{A}_S$ outputs -1 and since the error probability of the algorithm is at most ε, we know that (a) if $x \in P^{-1}(-1)$, then $R(x) \geq 1 - \varepsilon$, and (b) if $x \in P^{-1}(1)$, then $R(x) \leq \varepsilon$. If this is the case (a), we obtain $(1 - 2R(x)) \cdot P(x) = (1 - 2R(x)) \cdot (-1) \geq 1 - 2\varepsilon$. Moreover, even in case (b), we obtain $(1 - 2R(x)) \cdot P(x) = (1 - 2R(x)) \cdot (+1) \geq (1 - 2\varepsilon)$. Hence Equation 1 holds.

(1) using the assumption that $\hat{\mathrm{Q}}([N]) = 0$, (2) is argued using Cauchy-Schwarz inequality and (3) comes by using Parseval's theorem (on Boolean output and bounded functions). However, from Equation 1 we have that $\forall x \in S, Q(x) \cdot \mathrm{P}(x) \geq \varepsilon'$ which means

$$\begin{aligned}
\left(\sum_{B\in 2^{[N]}} \hat{\mathrm{Q}}(B)\cdot\hat{\mathrm{P}}(B)\right)^2 &= \left(\frac{1}{2^N}\sum_{x\in\{0,1\}^N} Q(x)\cdot \mathrm{P}(x)\right)^2 \\
&= \frac{1}{2^N}\left(\sum_{x\in S} Q(x)\cdot\mathrm{P}(x) + \sum_{x\in\{0,1\}^N\setminus S} Q(x)\cdot\mathrm{P}(x)\right)^2 \\
&\geq \left(\varepsilon'\cdot\frac{|S|}{2^N} + \frac{1}{2^N}\sum_{x\in\{0,1\}^N\setminus S} Q(x)\cdot\mathrm{P}(x)\right)^2 \\
&\geq \left(\varepsilon'\cdot\frac{|S|}{2^N} - \frac{2^N-|S|}{2^N}\right)^2 \\
&\geq \left(\left(1-\frac{1}{N^2}\right)\cdot\frac{|S|}{2^N} - \frac{2^N-|S|}{2^N}\right)^2 \\
&\geq \left(1-\frac{3}{N^2}\right)^2 \\
&> 1-\frac{6}{N^2},
\end{aligned}$$

leading to a contradiction; therefore, $\deg(Q(x)) = \deg(R(x)) = N$; using polynomial method [10] this implies $T = \Omega(N)$. ◀

Having stated Lemma 39 we can now immediately prove similar query lower bounds for PARITY and a *variant* of MAJORITY as their N-degree Fourier coefficients are 1 and at least $\sqrt{\frac{2}{\pi N}}$, respectively [27]. Moreover, this kind of result can be extended to show similar lower bounds for all Boolean properties P that have a non-negligible Fourier-mass of high degree coefficients. In fact, for PARITY we can show that similar lower bounds hold even on smaller sets as well. Also, we can prove such query lower bounds for MAJORITY and *st*-MAJORITY. More precisely,

▷ Claim 40 (see Lemma 43 in Appendix A for proof). Let $S \in \{0,1\}^N$ be a subset of size $s \geq 0.8\cdot 2^N$. Every quantum algorithm that, with success probability at least $\geq 2/3$, computes PARITY on each $x \in S$ uses $\Omega(N)$ queries.

and,

▷ Claim 41 (see Lemma 48 in Appendix A for proof). Let $S \in \{0,1\}^N$ be a subset of size $s \geq 2^N\left(1-\frac{1}{2\cdot N^2}\right)$. Every quantum algorithm that, with success probability $(1-\varepsilon) \geq (1-\frac{1}{2\cdot N^2})$, computes MAJORITY (*st*-MAJORITY) on each $x \in S$ uses $\Omega(N)$ queries.

The result proved in Lemma 28 when combined with the results stated in Claims 40 and 41 along with Theorem 34 lead us to the following result.

▶ **Corollary 42.** *Let $\Lambda = \mathsf{P/poly}$ be the complexity class. The properties PARITY $\in \mathcal{CO}(\Lambda)$, MAJORITY $\in \mathcal{CO}(\Lambda)$ and st-MAJORITY $\in \mathcal{CO}(\Lambda)$ unless Conjecture 33 is false.*

References

1 Scott Aaronson, Nai-Hui Chia, Han-Hsuan Lin, Chunhao Wang, and Ruizhe Zhang. On the quantum complexity of closest pair and related problems. In *Proceedings of the 35th Computational Complexity Conference*, CCC '20, Dagstuhl, DEU, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2020.16.

2 Amir Abboud, Thomas Dueholm Hansen, Virginia Vassilevska Williams, and Ryan Williams. Simulating branching programs with edit distance and friends: or: a polylog shaved is a lower bound made. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 375–388. ACM, 2016. doi:10.1145/2897518.2897653.

3 Andris Ambainis. A note on quantum black-box complexity of almost all boolean functions. *Inf. Process. Lett.*, 71(1):5–7, 1999. doi:10.1016/S0020-0190(99)00079-4.

4 Andris Ambainis, Kazuo Iwama, Akinori Kawachi, Hiroyuki Masuda, Raymond H. Putra, and Shigeru Yamashita. Quantum identification of boolean oracles. In Volker Diekert and Michel Habib, editors, *STACS 2004, 21st Annual Symposium on Theoretical Aspects of Computer Science, Montpellier, France, March 25-27, 2004, Proceedings*, volume 2996 of *Lecture Notes in Computer Science*, pages 105–116. Springer, 2004. doi:10.1007/978-3-540-24749-4_10.

5 Andris Ambainis, Kazuo Iwama, Akinori Kawachi, Rudy Raymond, and Shigeru Yamashita. Improved algorithms for quantum identification of boolean oracles. *Theor. Comput. Sci.*, 378(1):41–53, 2007. doi:10.1016/J.TCS.2006.12.013.

6 Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009. URL: http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521424264.

7 Srinivasan Arunachalam, Alex B. Grilo, Tom Gur, Igor C. Oliveira, and Aarthi Sundaram. Quantum learning algorithms imply circuit lower bounds. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 562–573. IEEE, 2021. doi:10.1109/FOCS52979.2021.00062.

8 Arturs Backurs and Piotr Indyk. Edit distance cannot be computed in strongly subquadratic time (unless SETH is false). In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 51–58. ACM, 2015. doi:10.1145/2746539.2746612.

9 Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 719–737. Springer, 2012. doi:10.1007/978-3-642-29011-4_42.

10 Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, 2001. doi:10.1145/502090.502097.

11 Harry Buhrman, Subhasree Patro, and Florian Speelman. A Framework of Quantum Strong Exponential-Time Hypotheses. In Markus Bläser and Benjamin Monmege, editors, *38th International Symposium on Theoretical Aspects of Computer Science (STACS 2021)*, volume 187 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 19:1–19:19, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.STACS.2021.19.

12 Yanlin Chen. *On Quantum Algorithms and Limitations for Convex Optimization and Lattice Problems*. PhD thesis, University of Amsterdam, 2024.

13 Yanlin Chen, Yilei Chen, Rajendra Kumar, Subhasree Patro, and Florian Speelman. Fine-grained complexity via quantum natural proofs. arXiv:2504.10363, 2025. doi:10.48550/arXiv.2504.10363.

14 Yanlin Chen, Yilei Chen, Rajendra Kumar, Subhasree Patro, and Florian Speelman. QSETH Strikes Again: Finer Quantum Lower Bounds for Lattice Problem, Strong Simulation, Hitting Set Problem, and More. In Alina Ene and Eshan Chattopadhyay, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2025)*, volume 353 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 6:1–6:24, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX/RANDOM.2025.6.

15 Nai-Hui Chia, Chi-Ning Chou, Jiayu Zhang, and Ruizhe Zhang. Quantum Meets the Minimum Circuit Size Problem. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, volume 215 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 47:1–47:16, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2022.47.

16 Timothy Y. Chow. Almost-natural proofs. *Journal of Computer and System Sciences*, 77(4):728–737, 2011. doi:10.1016/J.JCSS.2010.06.017.

17 Marek Cygan, Holger Dell, Daniel Lokshtanov, Dániel Marx, Jesper Nederlof, Yoshio Okamoto, Ramamohan Paturi, Saket Saurabh, and Magnus Wahlström. On problems as hard as cnf-sat. *ACM Transactions on Algorithms (TALG)*, 12(3):1–24, 2016. doi:10.1145/2925416.

18 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986. doi:10.1145/6490.6503.

19 Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996. doi:10.1145/237814.237866.

20 Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999. doi:10.1137/S0097539793244708.

21 Cupjin Huang, Michael Newman, and Mario Szegedy. Explicit lower bounds on strong quantum simulation. *IEEE Transactions on Information Theory*, 66(9):5585–5600, 2020. doi:10.1109/TIT.2020.3004427.

22 Russell Impagliazzo, Valentine Kabanets, Antonina Kolokolova, Pierre McKenzie, and Shadab Romani. Does looking inside a circuit help? In *42nd International Symposium on Mathematical Foundations of Computer Science*, volume 83 of *LIPIcs*, pages 1:1–1:13, 2017. doi:10.4230/LIPIcs.MFCS.2017.1.

23 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *Journal of Computer and System Sciences*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

24 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001. doi:10.1006/jcss.2001.1774.

25 Robin Kothari. An optimal quantum algorithm for the oracle identification problem. In Ernst W. Mayr and Natacha Portier, editors, *31st International Symposium on Theoretical Aspects of Computer Science (STACS 2014), STACS 2014, March 5-8, 2014, Lyon, France*, volume 25 of *LIPIcs*, pages 482–493. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2014. doi:10.4230/LIPIcs.STACS.2014.482.

26 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, fourier transform, and learnability. *Journal of ACM*, 40(3):607–620, 1993. doi:10.1145/174130.174138.

27 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2021. arXiv:2105.10386. arXiv:2105.10386.

28 Alexander A. Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997. doi:10.1006/JCSS.1997.1494.

29 Aviad Rubinstein. Quantum DNA sequencing & the ultimate hardness hypothesis. Blog post, 2019. URL: https://theorydish.blog/2019/12/09/quantum-dna-sequencing-the-ultimate-hardness-hypothesis/.

30 R. Ryan Williams. Self-improvement for circuit-analysis problems. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 1374–1385. ACM, 2024. doi:10.1145/3618260.3649723.

31 Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoretical Computer Science*, 348(2-3):357–365, 2005. doi:10.1016/J.TCS.2005.09.023.

32 Mark Zhandry. How to construct quantum random functions. In *FOCS*, pages 679–687. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.37.

A Proofs of Claim 40 and Claim 41

▶ **Lemma 43.** *Let $S \in \{0,1\}^N$ be a subset of size $s \geq 0.8 \cdot 2^N$. Every quantum algorithm that, with success probability at least $\geq 2/3$, computes* PARITY *on each $x \in S$ uses $\Omega(N)$ queries.*

Proof. Suppose there exists a T-query bounded-error quantum algorithm $\mathcal{A}_S$ to compute parity on each $x \in S$ with worst-case error probability $\leq 1/3$. Here, without loss of generality, we assume the output of $\mathcal{A}_S$ is $\in \{-1, 1\}$, where 1 means odd parity and -1 means even parity. Let $P(x)$ be the polynomial that represents $\mathcal{A}_S$'s probability of outputting -1 on input x. Since $\mathcal{A}_S$ has worst-case error probability at most $1/3$ on each $x \in S$, we can immediately see $(1-2P(x)) \cdot (-1)^{|x|} \geq 1/3$ for every $x \in S$, where $|x|$ denotes the Hamming weight of x.

Observe that $\deg(P(x)) = \deg(1-2P(x))$ and $\mathbb{E}_{x\in\{0,1\}^N}[(1-2P(x))(-1)^{|x|}]$ is the Fourier coefficient of the degree-N term of $(1-2P(x))$. We obtain

$$\begin{aligned}\mathbb{E}_{x\in\{0,1\}^N}[(1-2P(x))(-1)^{|x|}] =& \frac{1}{2^N}\Big(\sum_{x\in S}[(1-2P(x))(-1)^{|x|}] + \sum_{x\in\{0,1\}^N\setminus S}[(1-2P(x))(-1)^{|x|}]\Big)\\ \geq& \frac{1}{2^N}\cdot\big(s/3 - (2^N - s)\big) > 0,\end{aligned}$$

implying that $\deg(P(x)) \geq N$. Then by polynomial method [10], [13] we immediately obtain $T \geq N/2$. ◀

▶ **Lemma 44** (Theorem 5.19 in [27])**.** *Let N be an odd integer and let odd-*MAJORITY *: $\{0,1\}^N \to \{0,1\}$ be defined as*

$$\textit{odd-MAJORITY}(x) = \begin{cases} 1, & \text{if } |x| > \frac{N}{2},\\ 0, & \text{otherwise,}\end{cases}$$

*where $|x| = |\{i \mid i \in [N] \text{ and } x_i = 1\}|$. If the Fourier expansion of odd-*MAJORITY *is*

$$\textit{odd-MAJORITY}(x) = \sum_{B\subseteq\{0,1\}^N} \widehat{\textit{ODD-MAJ}}(B)\chi_B(x),$$

where $\chi_B(x) = (-1)^{x.B}$, then $\widehat{\textit{ODD-MAJ}}(B) = 0$ whenever $|B| == 0 \mod 2$ and for $|B| = N$ we have

$$\widehat{\textit{ODD-MAJ}}(B)^2 = \frac{4}{4^N}\binom{N-1}{\frac{N-1}{2}}^2 \geq \frac{6}{\pi(3N-1)} > \frac{2}{\pi N}.$$

▶ **Lemma 45.** *Let N be an odd integer and let $S \in \{0,1\}^N$ be a subset of size $s \geq 2^N\left(1 - \frac{1}{N^2}\right)$. Every quantum algorithm that, with error probability $\varepsilon \leq \frac{1}{2\cdot N^2}$, computes odd-*MAJORITY *on each $x \in S$ uses $\Omega(N)$ queries.*

Proof. The absolute value of the N-degree Fourier coefficient for *odd-*MAJORITY is at least $\sqrt{\frac{2}{\pi N}}$ using Lemma 39 gives us the required proof. ◀

[13] By the polynomial method we know if the amplitudes of the final state are degree-N polynomials of x then we need to make at least $T \geq N/2$ queries.

▶ **Lemma 46.** *Let N be an even integer and let $S \in \{0,1\}^N$ be a subset of size $s \geq 2^N(1-\frac{1}{2\cdot N^2})$. Every quantum algorithm that, with error probability $\varepsilon \leq \frac{1}{2\cdot N^2}$, computes* MAJORITY *on each $x \in S$ uses $\Omega(N)$ queries.*

We embed the *odd*-MAJORITY into the MAJORITY and then argue lower bound for MAJORITY for sets of size at least $2^N(1-\frac{1}{2\cdot N^2})$ using the result obtained in Lemma 45.

Proof. Towards contradiction, let us assume that there is a set $B \subseteq \{0,1\}^N$ of size $b \geq 2^N(1-\frac{1}{2\cdot N^2})$ that uses at most $N^{0.999}$ queries to compute MAJORITY on all strings $y \in B$. The set $B = \{B_0 \bigcirc 0\} \sqcup \{B_1 \bigcirc 1\}$ can be viewed as two disjoint sets of strings ending with 0 and 1, respectively. Clearly $|B_1| \leq 2^{N-1}$ and by our assumption we have $|B| \geq 2^N(1-\frac{1}{2N^2})$, therefore, $|B_0| = |B| - |B_1| \geq 2^N(\frac{1}{2} - \frac{1}{2\cdot N^2})$. Set $S' = B_0$. The set S' is a subset of strings of N'-length strings where $N' = N-1$ and moreover N' is odd. Therefore, in terms of N' we have $|B_0| \geq 2^{N'+1}(\frac{1}{2} - \frac{1}{2\cdot(N'+1)^2}) = 2^{N'} - \frac{2^{N'}}{\frac{2}{2}\cdot(N'+1)^2} > 2^{N'}(1-\frac{1}{N'^2})$. As $S' = B_0$, computing MAJORITY on strings in $B_0 \bigcirc 0$ (or equivalently $S' \bigcirc 0$) would be equivalent to computing *odd*-MAJORITY on N'-length strings in S' of size $|S'| \geq 2^{N'}(1-\frac{1}{N'^2})$. But we know from Lemma 45 a sublinear query algorithm is not possible for these set sizes. Hence, proved. ◀

We can give query lower bound for *st*-MAJORITY by just setting $S' = B_1$ and running the same arguments mentioned in the proof of Lemma 46. Giving the following result for *st*-MAJORITY.

▶ **Lemma 47.** *Let N be an even integer and let $S \in \{0,1\}^N$ be a subset of size $s \geq 2^N(1-\frac{1}{2\cdot N^2})$. Every quantum algorithm that, with error probability $\varepsilon \leq \frac{1}{2\cdot N^2}$, computes st-MAJORITY on each $x \in S$ uses $\Omega(N)$ queries.*

When N is odd, all the three definitions of MAJORITY, *st*-MAJORITY and *odd*-MAJORITY are equivalent. Therefore, combining Lemmas 45–47 we get the following.

▶ **Lemma 48.** *Let $S \in \{0,1\}^N$ be a subset of size $s \geq 2^N\left(1-\frac{1}{2\cdot N^2}\right)$. Every quantum algorithm that, with success probability $(1-\varepsilon) \geq (1-\frac{1}{2\cdot N^2})$, computes MAJORITY (st-MAJORITY) on each $x \in S$ uses $\Omega(N)$ queries.*

Locality of Curve-Decoding and Improved Proximity Gaps

Rohan Goyal
Massachusetts Institute of Technology, Cambridge, MA, USA

Venkatesan Guruswami
University of California, Berkeley, CA, USA

Yihang Sun
Stanford University, CA, USA

Mary Wootters
Stanford University, CA, USA

Abstract

Proximity gaps are a property of error correcting codes that arise in the study of Interactive Oracle Proofs (IOPs) and Succinct Non-interactive Arguments of Zero Knowledge (SNARKs). Informally, we say that a code $C \subset \Sigma^n$ exhibits a *proximity gap* (with respect to degree-ℓ curves) if for any degree-ℓ curve $u(x) \in \Sigma^n$, either *every* point on $u(x)$ is close to C, or else *most* of them are far from C.

Recent work [16] has established near-optimal proximity gaps for many families of codes, including subspace design codes, as well as random ensembles like random linear codes, Reed-Solomon codes with random evaluation points, and Gallager's ensemble of LDPC codes. However, the parameters for these latter randomized ensembles are worse than the parameters for subspace design codes, and degrade as the degree ℓ increases.

In this work, we obtain improved proximity gaps for random ensembles of codes, including random linear codes, Reed-Solomon codes with random evaluation points, and Gallager's ensemble. Quantitatively, our results for these random ensembles match the results that [16] attained for subspace design codes. In fact, our techniques are a black-box transference from subspace design codes: Any progress on subspace design codes will automatically lead to analogous progress for these random ensembles.

To obtain our results, we extend the Local Coordinate-wise Linear (LCL) property framework developed in [9, 27] to a *row-span constrained* version. This allows us to cast *curve-decodability* – a property that implies proximity gaps – directly as an (row-span constrained) LCL property, and make use of that machinery. In contrast, because curve-decodability is not obviously a (vanilla) LCL property, prior work had worked with a proxy property instead, leading to the aforementioned parameter losses. In addition, we extend the framework to also show an equivalence theorem for Gallager's ensemble of random LDPC codes and random linear codes for our row-span constrained LCL properties.

2012 ACM Subject Classification Theory of computation → Error-correcting codes; Theory of computation → Interactive proof systems

Keywords and phrases Proximity gaps, random codes, curve decoding, local properties

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.61

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2607.08516` [18]

Funding *Rohan Goyal*: R.G. is supported by (Yael Tauman Kalai's) grant from Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-25-C-0300. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA).



© Rohan Goyal, Venkatesan Guruswami, Yihang Sun, and Mary Wootters;
licensed under Creative Commons License CC-BY 4.0



Venkatesan Guruswami: V.G. is supported by a Simons Investigator award, NSF grant CCF-2211972, and ONR grant N00014-24-1-2491.
Yihang Sun: Y.S. is funded by the NSF Graduate Research Fellowship and the Stanford Graduate Fellowship.
Mary Wootters: M.W. is partially funded by NSF grants CCF-2231157 and CNS-2321489.

Acknowledgements The authors would like to thank Dan Boneh for helpful conversations.

1 Introduction

Proximity gaps are a property of error correcting codes that arise in the study of Interactive Oracle Proofs (IOPs) and Succinct Non-interactive Arguments of Zero Knowledge (SNARKs), both of which are relevant for blockchains and related technologies. Let $\mathbb{F}$ be a finite field, and let Σ be a vector space over $\mathbb{F}$. An $\mathbb{F}$-*additive code* C (with alphabet Σ and block length n) is an $\mathbb{F}$-linear subspace $C \subset \Sigma^n$. Informally, we say that C exhibits a *proximity gap* (for degree-ℓ curves, which is the focus of this paper) if the following holds. For any degree-ℓ curve $u(x) = \sum_{i=0}^{\ell} u_i x^i$ in Σ^n given by $u_0, \ldots, u_\ell \in \Sigma^n$ and $x \in \mathbb{F}$, either *every* point $u(\alpha)$ on the curve is close to C, or else a $1-\varepsilon$ fraction of the points on the curve are far from C. Formally, we have the following definition. Below, $\Delta(x,y) := \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}[x_i \neq y_i]$ is the (relative) Hamming distance between $x, y \in \Sigma^n$, and $\Delta(x, C) := \min_{y \in C} \Delta(x,y)$ for code C.

▶ **Definition 1** (Proximity Gaps (PG)). *An $\mathbb{F}$-additive code $C \subset \Sigma^n$ is said to have $(\ell, \delta, \varepsilon, \gamma)$ proximity gap if for all $u_0, u_1, \cdots, u_\ell \in \Sigma^n$, and for all $\delta' \leq \delta$,*

$$\mathbb{P}_{\alpha \in \mathbb{F}}\left(\Delta\left(\sum_{j=0}^{\ell} u_j \alpha^j, C\right) \leq \delta'\right) > \varepsilon \implies \forall \alpha \in \mathbb{F},\ \Delta\left(\sum_{j=0}^{\ell} u_j \alpha^j, C\right) \leq \frac{\delta'}{1-\gamma}\ .$$

When $\gamma = 0$, we say that C has an $(\ell, \delta, \varepsilon)$ proximity gap.

Proximity gaps, especially for *Reed-Solomon (RS) Codes*,[1] have been a hot topic lately, with the Ethereum Foundation offering $1M in prizes for resolving (or disproving) conjectures related to proximity gaps for RS codes [14]. See [2] for a recent survey on proximity gaps and problems relevant to the proximity prize.

Until recently, the best proximity gap guarantees required $\delta < 1 - \sqrt{R}$, where $R = \frac{\log_{|\Sigma|} |C|}{n}$ is the *rate* of the code $C \subset \Sigma^n$. In recent work [16], Guruswami and Goyal established near-optimal proximity gaps – with δ approaching $1 - R$ – for several families of codes. For *Folded Reed-Solomon Codes, Multiplicity Codes*,[2] and more generally any *Subspace Design Code* (Definition 15), they established $(\ell, \delta, \varepsilon)$ proximity gaps with $\delta = 1 - R - 2\eta$ for any $\eta > 0$, provided that

$$\varepsilon q \geq \frac{n\ell}{\eta} + O\left(\frac{\ell^2}{\eta^3}\right), \tag{1}$$

[1] Reed-Solomon codes are a classical family of codes based on low-degree polynomials. Given fixed distinct evaluation points $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q =: \Sigma$, the corresponding Reed-Solomon code $C \subset \Sigma^n$ of length n and dimension $k \leq n$ is given by $C = \{(f(\alpha_1), \ldots, f(\alpha_n)) : f \in \mathbb{F}_q[x], \deg(f) < k\}$.

[2] Informally, a *folded Reed-Solomon code* is obtained by "folding" (aka, bundling symbols together) an RS code $C \in \mathbb{F}_q^n$ to obtain a code $C' \in (\mathbb{F}_q^m)^{n/m}$. A (univariate) *multiplicity code* is similar to an RS code, except each symbol contains not just $f(\alpha_i)$, but also $f^{(j)}(\alpha_i)$ for $j = 1, 2, \ldots, m-1$, where $f^{(j)}$ denotes the j'th (Hasse) derivative. In either case, the alphabet size increases from q to q^m. By a result of [20], both are special cases of Subspace Design Codes, defined in Definition 15.

where q is the size of the base field. Thus, the left-hand side εq is a bound on the number of points on the curve $u(x)$ that can still be close to C, in the case that not all q points are. We'd like for this bound to be as small as possible.

Via a framework from [9, 27] that establishes a connection between random linear codes and subspace design codes, the work [16] extended their results for subspace design codes to several random ensembles of codes, including random linear codes, Reed-Solomon codes with random evaluation points, and Gallager's ensemble of random LDPC codes. However, some parameters degraded in this extension: The final result that [16] established for these random ensembles was $(\ell, \delta, \varepsilon)$ proximity gaps with $\delta = 1 - R - 2\eta$, provided that

$$\varepsilon q \geq \frac{n\ell(1-R)}{\eta} + \left(\frac{\ell}{\eta^2}\right)^{O(\ell)}. \tag{2}$$

When ℓ is large, on the order of $\Omega(\log n)$, the error term $(\ell/\eta^2)^{O(\ell)}$ in (2) swamps the main term, and the requirement on εq quickly becomes large. This leaves us with the question of whether a bound like (1) is achievable for these random ensembles of codes. This question has practical as well as theoretical motivations: Larger values of ℓ can occur in practice, for example in WHIR [3]; random ensembles of codes are also practically interesting, for example in Blaze [10] or Bolt [19];[3] and understanding RS codes with random evaluation points may deepen our understanding of explicit RS codes, relevant to the proximity prize [14].

1.1 Our Contributions

Our main contribution is to improve the bound (2) for random ensembles of codes to match the bound (1) that was already established for subspace design codes. We establish these improved bounds for random linear codes, RS codes with random evaluation points, and Gallager's ensemble of LDPC codes. See Definitions 5–7 for formal definitions of these codes.

▶ **Theorem 2** (Informal). *Fix any $\ell \in \mathbb{N}$ and $\eta > 0$. Let $C \in \mathbb{F}_q^n$ be a random linear code, random Reed-Solomon code, or random LDPC code*[4] *Then, with probability at least $2/3$ over the choice of C, C has $(\ell, 1 - R - 2\eta, \varepsilon)$ proximity gaps (with respect to degree-ℓ curves), provided that*

$$\varepsilon q \geq \frac{n\ell(1-R)}{\eta} + O\left(\frac{\ell^2}{\eta^3}\right). \tag{3}$$

In fact, our results establish not just proximity gaps, but also stronger notions known as *Correlated Agreement* (CA, Definition 8) and *Mutual Correlated Agreement* (MCA, Definition 9); see the full version for the full statements [18].

Our framework, discussed more below in Section 1.2, is a black-box transference method from subspace design codes. Thus, any future improvements for subspace design codes will immediately lead to an analogous improvement in Theorem 2.

In order to establish our results, we generalize the *Local Coordinate-wise Linear* (LCL) framework of [27] to a new "row-constrained" version of the framework, described below. We hope that this generalization may find other applications going forward.

[3] In more detail, Blaze considers a random ensemble of *RAA* codes, which use random permutations to decide the parity checks. Bolt uses *Gallager's ensemble* of LDPC codes, which we also study in this paper.

[4] Technically, the result for random LDPC codes requires the leading term $n\ell(1-R)/\eta$ to be slightly larger; see full version for details.

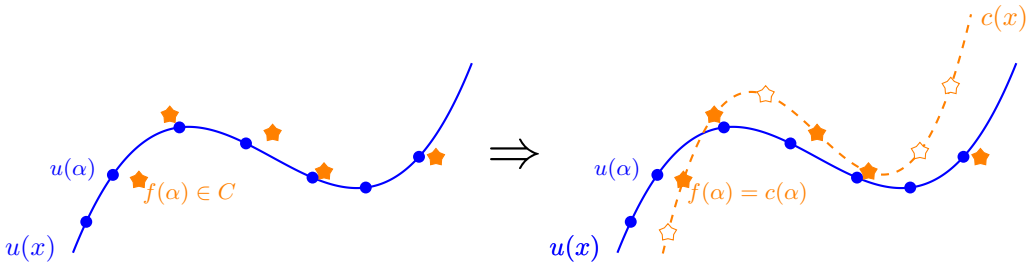


Figure 1 Depiction of curve decodability (Definition 10). The orange stars represent $f(\alpha) \in C$ for $\alpha \in A$, and the blue dots represent $u(\alpha) \in \Sigma^n$ for $\alpha \in \mathbb{F}_q$, where $u(x)$ is a degree-ℓ curve. Informally, a code $C \subseteq \Sigma^n$ is (ℓ, δ, a, b)-curve-decodable if the following holds: If there are at least a orange stars δ-close to distinct blue dots, then at least b of those orange stars lie on a degree-ℓ curve $c(x)$. In the picture, $a = 5$, and $b = 4$.

▶ Remark 3. [27] showed an equivalence for LCL properties between random linear and random Reed-Solomon codes; we state its row-span constrained extension here. The analogous statement for Gallager's ensemble, based on [28], is recorded in Section 3.3.3; the proofs and LDPC-specific details appear in the full version [18].

1.2 Techniques: Curve-Decoding as a Row-Span Constrained LCL Property

Following [16], we study *curve-decoding*. For the rest of this discussion, assume for simplicity that the alphabet Σ is equal to $\mathbb{F}_q$, so C is a linear subspace of $\mathbb{F}_q^n$.

Informally, such a code C is (ℓ, δ, a, b) *curve-decodable* if, for any degree-ℓ curve $u(x) = \sum_{i=0}^{\ell} u_i x^i$ with $u_i \in \Sigma^n$, the following holds. Suppose that there is a set $A \subset \mathbb{F}_q$ of size a so that $u(\alpha)$ is δ-close to some codeword $f(\alpha) \in C$ for all $\alpha \in A$. Then, there should be a "codeword curve" $c(x) = \sum_{i=0}^{\ell} c_i x^i$, where $c_i \in C \subset \Sigma^n$, so that $f(\alpha) = c(\alpha)$ for at least b values of $\alpha \in A$. See Definition 10 for a formal definition; the property is illustrated in Figure 1.

The work [16] showed that curve-decodability is a sufficient condition to establish proximity gaps (as well as stronger notions like correlated agreement and mutual correlated agreement); see Theorem 12 and Theorem 13. They then established that subspace design codes have curve-decodability (Theorem 18), which led to their results about subspace design codes.

Our work comes in the next step. To obtain their results for random ensembles of codes, [16] applied a framework of [9, 27]. This framework roughly shows that any *Local Coordinate-wise Linear (LCL)* property that is satisfied by subspace design codes is also satisfied by these random ensembles. We will explain what LCL properties are in a moment, but the main challenge with this approach is that "being curve-decodable" is in fact *not* (or, not obviously) an LCL property. To get around this, [16] defined another property, which they called $\bigvee$*-decoding*, which *is* captured by the LCL framework. Then existing results imply that the random ensembles of codes are just as $\bigvee$-decodable as subspace design codes, and translating back to curve-decoding gives the final result. Unfortunately, going through $\bigvee$-decoding results in a loss in parameters, degrading (1) to (2). We avoid this loss by *extending* the LCL framework to apply directly to curve-decoding.

The LCL Framework. Before explaining our extension, we first explain the LCL framework of [27]. Intuitively, an r-local LCL property is one defined by the exclusion of r-sized bad sets of codewords. For example, the property of "having good distance" might be defined by the

exclusion of all pairs ($r = 2$) of codewords that agree in too many places. The LCL framework allows for any definition of "bad" that can be captured by a family $\mathcal{F}$ of coordinate-wise linear relationships.

In more detail, let $C \subset \mathbb{F}_q^n$ be a linear code, and consider a set of r distinct "bad" codewords, stacked together to form the columns of a matrix $M \in \mathbb{F}_q^{n \times r}$. Let $\mathcal{F}$ be a family of *profiles* $\mathbf{V} = (V_1, \ldots, V_n)$, where each $V_i \in \mathcal{L}(\mathbb{F}_q^r)$. (Here, $\mathcal{L}(W)$ denotes the collection of subspaces of W.) In the LCL framework, the "badness" of a matrix M is captured by the following criterion: There is some $\mathbf{V} = (V_1, \ldots, V_n) \in \mathcal{F}$ so that for all $i \in [n]$, the i'th row of M is contained in V_i.

Thus, an LCL property defined by $\mathcal{F}$ is the property of *avoiding* all "bad" matrices M that satisfy the constraints given by some $\mathbf{V} \in \mathcal{F}$ (we say that C *avoids* M if the column span of M is not contained in C). See Definition 20 for a formal definition.

Capturing Curve-Decoding and the Approach of [16]. The LCL framework captures many familiar properties, for example distance, list-decoding, and list-recovery. Unfortunately, it does not seem to capture curve-decoding. To see why, let us consider two approaches.

For the first – and probably most natural – approach, suppose that we have a bad example for (ℓ, δ, a, b) curve-decoding. That is, suppose we have a set $A \subset \mathbb{F}_q$ of size a, a degree-ℓ curve $u(x) \in \mathbb{F}_q^n$, and a function $f : \mathbb{F}_q \to C$ so that $\Delta(u(\alpha), f(\alpha)) \leq \delta$ for all $\alpha \in A$; but suppose that there is *no* degree-ℓ "codeword curve" $c(x) = \sum_{i=0}^{\ell} c_i x^i$ for $c_i \in C$ that passes through at least b of the $f(\alpha)$.

We will attempt to construct a "witness" matrix $M \in \mathbb{F}_q^{n \times A}$ whose columns are $f(\alpha)$ for all $\alpha \in A$. In order to certify this as a bad example, we should be able to come up with some LCL profile $\mathbf{V} = (V_1, \ldots, V_n)$ that "explains" the badness of the example. There are two properties that make a "bad example" bad:

(A) Each of the $f(\alpha)$ (the columns of M) agree in many coordinates with some ambient curve $u(x)$ of degree ℓ.

(B) No b columns of M themselves lie on the same degree-ℓ curve.

Item (A) can be captured with an a-local profile $\mathbf{V} = (V_1, \ldots, V_n)$. Indeed, let $A_i = \{\alpha \in A : u(\alpha)_i = f(\alpha)_i\}$, and then define V_i to be the subspace of all $y \in \mathbb{F}_q^A$ so that y (when viewed as a function $y : A \to \mathbb{F}_q$) agrees with some degree-ℓ polynomial on A_i. Then consider the class $\mathcal{F}$ of all profiles $\mathbf{V}$ generated this way (over all appropriate choices of A_i). Item (A) holds if and only if C contains some $\mathbf{V} \in \mathcal{F}$.

However, it is not clear how to capture item (B) with the LCL framework, as (B) is a more global property of the matrix, not easily captured with coordinate-wise constraints. Thus, another approach seems needed.

As a second possible approach, we describe what [16] did, and briefly explain why it incurs an exponential loss in the parameter ℓ. The idea is to consider first the special case that $b = \ell + 2$. Given a bad example for curve-decoding, consider a much larger "witness" matrix $M \in \mathbb{F}_q^{n \times r}$ for $r = \binom{a}{\ell+1}$, as follows. The columns of M now correspond to subsets $S \in \binom{A}{\ell+1}$, and the columns correspond to polynomials $g_S(x)$ defined by interpolating through the points $(\alpha, f(\alpha))$ for all $\alpha \in S$. It turns out that if $b = \ell + 2$, then all of these polynomials must be distinct: Otherwise, if $g_S = g_{S'}$ for some $S \neq S'$, then this gives a degree-ℓ curve passing through at least $|S \cup S'| \geq \ell + 2 = b$ of the codewords $f(\alpha)$, contradicting the badness of this example. Then, if $\mathbb{F}_q$ is large enough, there is some β so that $\left\{g_S(\beta) \in C : S \in \binom{A}{\ell+1}\right\}$ are also all distinct, and these can be the columns of M. However, this requires us to take $\ell = b + 2$; otherwise, we can't be guaranteed that the columns are distinct, which is required by the LCL framework. The insight of [16] was to essentially transform the general-ℓ case

to the $\ell = b + 2$ case via a new property called $\bigvee$-decoding; but this came at a cost of an exponential dependence on ℓ. (We emphasize that the exponential loss comes from the transformation to the $b = \ell + 2$ case, not the larger size of the witness matrix, although we will be able to overcome both issues in this work.)

Our Approach: Row-span Constrained LCL Properties. Instead of trying to manipulate curve-decodability to fit into an LCL property, we extend the definition of an LCL property. That is, we return to the first, more straightforward, attempt above. Recall that the first property (A) – that each column of M is close to some degree-ℓ curve $u(x)$ –is easily captured by an a-local LCL profile. The harder part is dealing with item (B), that no b columns of M should themselves lie on a degree-ℓ curve. This is a global constraint on the matrix M, and thus is not immediately amenable to the coordinate-wise framework of [27].

In order to capture item (B), we introduce a *row-span* constraint. One example of such a constraint was already present in prior works [9, 27]. Indeed, that framework requires that the columns of the witness matrix M be distinct, which turns out to be equivalent to the property that the row-span U of M does not have any identically equal coordinates (that is, there should not be any $i \neq j$ so that $u_j = u_i$ for all $u \in U$). Thus, [9, 27] restricted the row-span of their witness matrices M to the set $\mathcal{L}_{\text{dist}}(\mathbb{F}_q^r)$ of *distinct* subspaces U, those without any identically equal coordinates.

Our observation is that item (B) can also be captured in this way. In Lemma 50 and Corollary 51, we show that item (B) is equivalent to the following: For any $B \in \binom{A}{b}$, there is some u in the row-span U of M so that $u|_B$ does *not* agree with any degree-ℓ curve $p : B \to \mathbb{F}_q$. We define a subspace U with this property to be (ℓ, A, b) *curve-free* (Definition 49). Thus, in addition to an a-local LCL profile $\mathbf{V}$ to enforce the condition (A), we also have a condition on the row-span – namely that the row-span U of M should be (ℓ, A, b) curve free – to enforce condition (B).

Formally, we implement this by defining the set $\mathcal{F}$ to be a set of *pairs* $(\mathbf{V}, U)$, where $\mathbf{V}$ is an r-local profile, and $U \subset \mathbb{F}_q^r$ is a subspace. Then a witness matrix M with columns in C is "bad" if there is some $(\mathbf{V}, U) \in \mathcal{F}$ so that the i-th row of M is contained in V_i for all i; and additionally the row-span of M is contained in U. We say that C *avoids* $\mathcal{F}$ if there is no such witness matrix.

Putting it Together. In order to implement the ideas above, we first re-derive the LCL framework and the connections in [9, 27] in the row-span constrained setting. We do this in Section 3. To do so, in Section 3, we introduces row-span constrained LCL profiles and states the corresponding threshold and transfer theorems for random linear, random Reed–Solomon, random LDPC, and subspace-design codes. In particular, Theorems 34 and 45 and Corollaries 37 and 48 identify the relevant threshold rate and its transfer to subspace-design codes.

Next, in Section 4, we characterize *not* being (ℓ, δ, a, b) curve-decoding as a row-span constrained a-local LCL property, as described above. The local profile records the agreement of the nearby codewords with an ambient degree-ℓ curve (property (A)), while the row-span constraint records the absence of a degree-ℓ codeword curve through any b nearby codewords (property (B)). This is stated formally in Theorem 52 and Corollary 53.

Finally, we apply our row-span constrained LCL theory to curve-decoding and to three random ensembles of codes – random linear codes, RS codes with random evaluation points, and Gallager's ensemble of LDPC codes – to prove Theorem 2. To recap, we know from [16] that subspace design codes satisfy curve-decodability; we have shown in Section 3 that

random ensembles of codes behave similarly to subspace design codes for any *row-span constrained* LCL property; and we have shown in Section 4 that curve-decodability is such a property. Therefore random ensembles of codes exhibit good curve-decodability as well. Section 5 states the resulting proximity-gap, CA, and MCA guarantees for the three random ensembles.

Due to space constraints, we omit some proofs and calculations; the reader can find these in the full version of this paper [18].

1.3 Related Works

Proximity Gaps. Proximity gaps (and related notions, including correlated agreement and mutual correlated agreement) arise naturally in the study of interactive protocols. The goal in these settings is to test whether or not *every* vector in a set – say, a line or a curve – is close to a code C. If that code exhibits a proximity gap, then it suffices to choose a random vector in that set and use its distance from the code as a proxy for the minimum distance to C over the whole set. This connection was observed in [29], and since then a long line of work has been devoted to improving the parameters [1, 3, 6–8, 16] and establishing limitations [5, 12, 13, 26]. See the recent survey [2] for an overview of the work on proximity gaps and related notions. Prior to the work of [16], the best known proximity gap results required $\delta < 1 - \sqrt{R}$, corresponding to the *Johnson radius* in list-decoding. As discussed above, the work [16] improved this to allow δ to approach $1 - R$, the best trade-off possible. However, as noted earlier, the results can still be improved, notably in how small the parameter ε can be. The current best results for explicit codes – including folded RS codes, multiplicity codes, and all subspace design codes – follow from [16] and work under the condition given by (1). In this work we improve their results for random ensembles of codes from (2) to (1).

Local properties and subspace design codes. A recent line of work [21, 22, 27, 28] has shown that a wide variety of random ensembles of codes – including random linear codes, random LDPC codes, and Reed-Solomon codes with random evaluation points – all behave similarly with respect to "local" properties: properties that are characterized by the exclusion of small bad sets. The paper [27] introduced the LCL framework discussed above, and [9] connected this to another line of work on *subspace design codes*. The study of subspace design codes implicitly began with the introduction of Folded Reed-Solomon codes [23]. Over nearly two decades, it was observed that a key property of these codes, as well as of univariate multiplicity codes, is that they satisfy a *subspace design property* [20], which has been leveraged and generalized in subsequent works [4, 9, 11, 17, 25, 30, 31]. Notably, [11] showed that *any* such subspace design codes enjoy optimal list-decodability. The work [9] then connected this line of work to the LCL framework discussed above, showing that in fact the same thresholds are achieved by subspace design codes. The work [16] leveraged this connection to study curve-decoding, and we do the same in this work.

1.4 Discussion and Open Problems

We conclude this section with a few open questions.

- Our approach can automatically turn any improvements for subspace design codes into analogous improvements for random ensembles of codes. Thus, the natural open question is to further improve the trade-offs between ε, q, ℓ and η for subspace design codes.

- Our results imply that Reed-Solomon codes with random evaluation points attain near-optimal proximity gaps, even for large ℓ. While it is known that there exist Reed-Solomon codes that do *not* achieve optimal proximity gaps [5, 12, 13, 26], it is not known which choices of evaluation points are "good." Can we identify explicit Reed-Solomon codes with near-optimal proximity gaps? This question is related to the open problem of identifying explicit Reed-Solomon codes that achieve near-optimal list-decoding.
- We hope that our techniques will be useful beyond the present application. Are there other natural properties that can be captured with the row-span constrained LCL framework, but not with the original LCL framework?

2 Preliminaries

We begin by introducing the basic coding-theoretic definitions we will be using. For any two vectors x, y in Σ^n where Σ is some alphabet, we define the fractional Hamming distance $\Delta(x,y) = \frac{1}{n}|\{i \in [n] : x_i \neq y_i\}|$ to be the fraction of coordinates where they differ. For a set $S \subset \Sigma^n$, we define $\Delta(x,S) = \min_{y \in S} \Delta(x,y)$ to be the fractional distance of x to its closest vector in S. Throughout this paper, all distances are taken to be fractional unless stated otherwise.

The two fundamental quantities associated with a code are its rate and distance. For a code $C \subset \Sigma^n$, we define its (relative) distance as $\delta(C) = \min_{x,y \in C, x \neq y} \Delta(x,y)$. Moreover, its rate $R(C)$ is defined as $R(C) = \frac{1}{n}\log_{|\Sigma|}|C|$.

In this paper, we will focus on additive codes over a finite field, defined as follows:

▶ **Definition 4** (Additive codes). *Let $\mathbb{F}$ be a finite field and let $\Sigma = \mathbb{F}^s$ for some positive integer s. A code $C \subset \Sigma^n$ is said to be $\mathbb{F}$-additive (or just additive when the field $\mathbb{F}$ is clear from context) if C is an $\mathbb{F}$-linear subspace of Σ^n.*

We focus on three families of random codes; the first is a random linear code.

▶ **Definition 5** (Random Linear Code (RLC) and Random $\mathbb{F}_q$-Additive Code). *A random linear code $C \subset \mathbb{F}_q^n$ of rate R is given by taking $C = \{(\mathsf{Enc}_1(x), \cdots, \mathsf{Enc}_n(x)) \mid x \in \mathbb{F}_q^{Rn}\}$ where $\mathsf{Enc}_i : \mathbb{F}_q^{Rn} \to \mathbb{F}_q$ are independent uniformly random linear maps.* [5] *More generally, a random $\mathbb{F}_q$-additive code $C \subset (\mathbb{F}_q^s)^n$ of rate R is given by taking $C = \{(\mathsf{Enc}_1(x), \ldots, \mathsf{Enc}_n(x)) \mid x \in \mathbb{F}_q^{Rsn}\}$ where $\mathsf{Enc}_i : \mathbb{F}_q^{Rsn} \to \mathbb{F}_q^s$ are independent uniformly random linear maps.*

The next family is a random Reed-Solomon code, which is just a Reed-Solomon code where the evaluation points are chosen uniformly at random.

▶ **Definition 6** (Random Reed-Solomon Code (RRS)). *A Reed-Solomon code over a field $\mathbb{F}_q$ with evaluation points $\alpha_1, \cdots, \alpha_n$ is defined as follows*

$$\mathsf{RS}_{\mathbb{F}_q}(\alpha_1, \ldots, \alpha_n, k) = \{(f(\alpha_1), \ldots, f(\alpha_n)) \mid f \in \mathbb{F}_q[X], \deg f < k\}$$

A random Reed-Solomon code of rate R is one in which $k = Rn$ and α_i are independent uniformly random elements of $\mathbb{F}_q$.

[5] Note that, defined this way, there is some small chance that the rate $\log_q |C|/n$ of a random linear code may not actually be R, if there are too many linear dependencies between the maps Enc_i. However, it is more convenient to work with i.i.d. encoding maps, and the probability of the rate not being R is negligible, so anything that holds with high probability for this model also holds with high probability for other models of random linear codes, for example the model where one chooses C to be a uniformly random subspace of dimension Rn, or the model where C is the kernel of a random matrix in $\mathbb{F}_q^{(1-R)n \times n}$. See, e.g., [27, Appendix A].

Finally, we introduce our last family of codes, random Low-Density Parity-Check (LDPC) codes. In this paper, by "random LDPC codes" we mean *Gallager's ensemble* of random LDPC codes [15]. In general, a rate R, s-sparse LDPC code can be described as follows: Pick a bipartite graph $G = (L \sqcup R, E)$ with $|L| = n$, $|R| = (1-R)n$ and right degree s. Each edge $e \in E$ has a weight $w_e \in \mathbb{F}_q$, the code C is given as follows:

$$C = \Big\{ c \in \mathbb{F}_q^L : \sum_{u \in N(v)} w_{(u,v)} c_u = 0 \text{ for all } v \in R \Big\}.$$

Above, $N(v)$ denotes the neighborhood of v in G.

To define a random LDPC code, we thus just need to describe the underlying bipartite graph and the edge weights w_e. Gallager's ensemble [15] defines an ensemble of random graphs (and hence random LDPC codes) as follows.[6]

▶ **Definition 7** (Random Low Density Parity Check Code (RLDPC)). *Let $t = (1-R)s$, and let $G_i = (V, W_i, E_i)$ for $i = 1, \cdots, t$ be uniformly random $(1, s)$ regular bipartite graphs with a shared left vertex set and disjoint right vertex sets W_i of size n/s. The graph G is then defined as $\bigsqcup_i G_i$; that is, $G = (V, W, E)$, where $W = \bigsqcup_i W_i$ and $E = \bigsqcup_i E_i$. The edge weights w_e are sampled as independent uniformly random elements of $\mathbb{F}_q^*$. A* Random LDPC Code *C of rate R is then the LDPC code derived from G and the weights $\{w_e\}$, as described above.*

2.1 Proximity Gaps and (Mutual) Correlated Agreement

Proximity Gaps are related to two stronger notions, *Correlated Agreement* (CA) and *Mutual Correlated Agreement* (MCA). Our results actually establish both CA and MCA as well as proximity gaps.

Informally, we say that a code C has $(\ell, \delta, \varepsilon, \gamma)$-correlated agreement if, for all degree-ℓ curves $u(x) = \sum_{i=0}^{\ell} u_i x^i$ with $u_i \in \Sigma^n$, the following holds: Either at most an ε fraction of the points on $u(x)$ are δ-close to C; or there is some "codeword curve" $c(x) = \sum_{i=0}^{\ell} c_i x^i$ with $c_i \in C$ so that $u(x)$ agrees identically with $c(x)$ on a set $S \subset [n]$ of at least $\delta n/(1-\gamma)$ coordinates. Formally, we have the following definition.

▶ **Definition 8** (Correlated Agreement (CA)). *An $\mathbb{F}$-additive code $C \subset \Sigma^n$ is said to have $(\ell, \delta, \varepsilon, \gamma)$ correlated agreement if for all $u_0, u_1, \cdots, u_\ell \in \Sigma^n$ and $\delta' \le \delta$*

$$\mathbb{P}_{\alpha \in \mathbb{F}}\Big[\Delta\big(\sum_{j=0}^{\ell} u_j \alpha^j, C\big) \le \delta'\Big] > \varepsilon \implies \exists c_0, c_1, \cdots c_\ell \in C : \Big|\Big\{ i : \bigvee_{j=0}^{\ell} c_{j,i} \neq u_{j,i} \Big\}\Big| \le \frac{\delta' n}{1-\gamma}.$$

In particular, CA implies that for all $\alpha \in \mathbb{F}$, we have that $\Delta\Big(\sum_{j=0}^{\ell} u_j \alpha^j, \sum_{j=0}^{\ell} c_j \alpha^j\Big) \le \delta'/(1-\gamma)$. Thus, $(\ell, \delta, \varepsilon, \gamma)$ correlated agreement implies an $(\ell, \delta, \varepsilon, \gamma)$proximity gap.

Next, we define Mutual Correlated Agreement (MCA). Informally, we say that an $\mathbb{F}$-additive code C has $(\ell, \delta, \varepsilon)$ MCA if for all degree-ℓ curves $u(x)$ in Σ^n, the following holds. For all sets $S \subset [n]$ of size at least $(1-\delta)n$, and for at least an $1-\varepsilon$ fraction of $\alpha \in \mathbb{F}$, if there is some $c \in C$ that agrees with $u(\alpha)$ on S, then there is some "codeword curve" $c(x) = \sum_{i=0}^{\ell} c_i x^i$ for $c_i \in C$ so that $c(x)$ and $u(x)$ agree identically on S. Formally, we have the following definition, which captures the contrapositive of the intuition above.

[6] Technically, this way of generalizing to larger alphabets is slightly different than [15]. Our definition coincides with [15] for $q = 2$; for larger q, [15] chooses weights w_e to all be 1, while we choose them uniformly in $\mathbb{F}_q^*$; this matches the ensemble studied in [28].

▶ **Definition 9** (Mutual Correlated Agreement (MCA)). *An $\mathbb{F}$-additive code $C \subset \Sigma^n$ is said to have $(\ell, \delta, \varepsilon)$ mutual correlated agreement if for all $u_0, u_1, \cdots, u_\ell \in \Sigma^n$ and $\delta' \leq \delta$*

$$\mathbb{P}_{\alpha \in \mathbb{F}}\left(\exists S \subset [n], c \in C, j \in \{0,1,\ldots,\ell\} : |S| \geq (1-\delta')n, c_{|S} = \sum_{j=0}^{\ell} \alpha^j u_{j|S}, c'_{|S} \neq u_{j|S} \forall c' \in C \right) \leq \varepsilon.$$

We note that $(\ell, \delta, \varepsilon)$ MCA implies $(\ell, \delta, \varepsilon, \gamma = 0)$ CA and hence $(\ell, \delta, \varepsilon, \gamma = 0)$ PG. Indeed, suppose that C satisfies MCA, and suppose there is some codeword $c \in C$ that is δ'-close to $u(x)$ on at least εq points, call them $A \subset \mathbb{F}$. MCA implies that there is a set $B \subset \mathbb{F}$ of size at least $(1-\varepsilon)q$ so that the event inside the $\mathbb{P}_\alpha[\cdot]$ statement holds. Then there must be at least one point $\alpha \in A \cap B$. Letting $S \subset [n]$ be the set of coordinates (of size at least $(1-\delta')n$) on which $u(\alpha)$ agrees with c, we conclude from the definition of MCA that there is some "codeword curve" $c(x)$ so that $c(x)$ and $u(x)$ identically agree on S, establishing CA.

Finally, we formally define curve-decodability, which we informally defined in Section 1 (see Figure 1). The following definition was given in [16], and the case of $\ell = 1$ was independently defined in [24] as *collinearity of proximates*. We adopt the convention from [16].

▶ **Definition 10** (Curve-Decodability). *An additive code $C \subset \Sigma^n$ is (ℓ, δ, a, b) curve-decodable if for every $u_0, u_1, \cdots, u_\ell \in \Sigma^n$, all functions $f : \mathbb{F}_q \to C$, whenever the set $A = \left\{\alpha \in \mathbb{F} : \Delta\left(\sum_{j=0}^{\ell} u_j \alpha^j, f(\alpha)\right) \leq \delta\right\}$ has at least a elements, there exist $c_0, c_1, \cdots, c_\ell \in C$ such that $\left|\left\{\alpha \in A : f(\alpha) = \sum_{j=0}^{\ell} c_j \alpha^j\right\}\right| \geq b$.*

The work [16] showed a transformation to boost results about the curve-decodability of a code assuming that the underlying code is also list-decodable.

▶ **Theorem 11** ([16, Theorem 3.6]). *Suppose an $\mathbb{F}_q$-additive code $C \subset (\mathbb{F}_q^s)^n$ is $\left(\delta\left(1+\frac{\ell}{b-\ell}\right), L\right)$-list-decodable and (ℓ, δ, a, b)-curve-decodable. Then C is also $(\ell, \delta, (T-1)L + a, T)$ curve-decodable for any T.*

The key motivation to define curve-decodability is due to the following implication for correlated and mutual correlated agreement.

▶ **Theorem 12** (Correlated agreement from curve-decodability, [16, Theorem 3.4]). *Let $C \subset \Sigma^n$ be an $\mathbb{F}_q$-additive code. Let $\ell, t > \ell, a \in \mathbb{N}$ and $\delta \in (0,1)$. Suppose that $\mathcal{C}$ is (ℓ, δ, a, t) curve-decodable. Then C has $(\ell, \delta, a/q, \ell/t)$ correlated agreement.*

▶ **Theorem 13** (Mutual correlated agreement from curve-decodability, [16, Theorem 3.5]). *Let $C \subset \Sigma^n$ be an $\mathbb{F}_q$-additive code. Let $\ell, a \in \mathbb{N}$ and $\delta \in (0,1)$. Suppose that $\mathcal{C}$ is $(\ell, \delta, a, \ell n + 1)$ curve-decodable. Then C has $(\ell, \delta, a/q)$ mutual correlated agreement.*

▶ **Corollary 14.** *If an $\mathbb{F}_q$-additive code $C \subset (\mathbb{F}_q^s)^n$ is $\left(\delta\left(1+\frac{\ell}{b-\ell}\right), L\right)$-list-decodable and (ℓ, δ, a, b)-curve-decodable, then C has:*

1. *for any $t \geq 1$, $(\ell, \delta, (t-1)L + a, t)$ curve-decodability,*
2. *for any $m > 1$, $\left(\ell, \delta, \frac{\ell m L + a}{q}, \frac{1}{m}\right)$ correlated agreement and proximity gap,*
3. *$(\ell, \delta, \frac{\ell n L + a}{q})$ mutual correlated agreement.*

Enabled by the above reductions, to prove an additive code C has correlated agreement, it suffices to prove list-decodability and curve-decodability of the code. In this work, and related works, we always work in the regimes where the codes in consideration are already known to be list-decodable with good parameters. Thus, the key innovation lies in establishing curve-decodability.

2.2 Subspace-Design Codes

We introduce notation and results from [16] that we will use.

▶ **Definition 15** (Subspace-Design Code). *For any function $\tau : \mathbb{N} \to \mathbb{R}_{\leq 1}$, an $\mathbb{F}_q$-additive code $C \subset (\mathbb{F}_q^s)^n$ is said to be a τ-subspace design code if for every $r \in \mathbb{N}$, and every $\mathbb{F}_q$-linear subspace A of C of dimension at most r, $\frac{1}{n}\sum_{i=1}^{n} \dim A_i \leq \dim(A) \cdot \tau(r)$, where $A_i = \{a \in A \mid a_i = 0\}$.*

▶ **Definition 16.** *For an additive code $C \subset \left(\mathbb{F}_q^s\right)^n$ and dimension r, let*

$$\sigma(r) := \max\left\{\mathbb{E}_{i\in[n]}\left[\frac{\dim V_i}{\dim V}\right] : V \subset C, \dim V = r\right\} \tag{4}$$

where $V_i = \{x \in V : x_i = 0\}$. In the message space with encoding maps $\mathrm{Enc}_i : \mathbb{F}_q^k \to \mathbb{F}_q^s$, we have

$$\sigma(r) := \max\left\{\mathbb{E}_{i\in[n]}\left[\frac{\dim(A \cap \ker(\mathrm{Enc}_i))}{\dim A}\right] : A \subset \mathbb{F}_q^k, \dim A = r\right\} \tag{5}$$

Following the notation of [16], we can define the monotone envelope $\tau(r) := \max\{\sigma(1), \ldots, \sigma(r)\}$. Observe that with this choice, C is by definition a τ-subspace design code.

▶ Remark 17 (Comparison to [9]). The paper [9] uses slightly different language; they say that C is *r-subspace designable if*, for every $d \in [r]$, $\sigma(d) \leq R + \frac{1}{nd}$. We recall that $\sigma(r) \geq R - r/sn$ always. Thus, having the subspace design property with "good" parameters means that we should have $\tau(d) \leq R + \eta$ for some "small" $\eta > 0$. [9] proved that random $\mathbb{F}_q$-additive codes $C \subset (\mathbb{F}_q^s)^n$ are r-subspace designable whenever $s = \Omega(rn)$.

Finally, we state a theorem from [16] which shows that any subspace-design code is curve-decodable.

▶ **Theorem 18** ([16, Theorem 4.7]). *Fix any integers a, ℓ and $\eta > 0$, and let $d = \lceil(\ell + 1)/\eta\rceil$. Then for any τ-subspace design code $C \subset (\mathbb{F}_q^s)^n$, C is $\left(\ell, 1 - \tau(d) - \eta, a, \frac{\eta a}{d+\eta}\right)$-curve-decodable.*

3 Row-span Constrained Local Properties

3.1 Generalizing Local Profiles

We begin by introducing some notation from [9, 27], and explaining how we generalize it to our row-span constrained versions. As described in Section 1.2, a *local profile* gives a set of coordinate-wise linear constraints that a "bad" witness matrix M might satisfy. This section states the threshold and transfer results needed later; their proofs appear in the full version [18].

▶ **Definition 19** (r-Local Profile, [27]). *Let $\mathbb{F}_q$ be a finite field of q elements and let n and r be positive integers. A tuple $\mathbf{V} = (V_1, \cdots, V_n) \in \mathcal{L}(\mathbb{F}_q^r)^n$ is called a r-local profile.*

Next, we define what it means for a code to *contain* a local profile (with a particular row-span).

▶ **Definition 20** (Containing a local profile, [27]). *A code $C \subset \mathbb{F}_q^n$ is said to* contain *a local profile $\mathbf{V}$ with span $U \in \mathcal{L}(\mathbb{F}_q^r)$ (often written "C* contains *$(\mathbf{V}, U)$") if there exists an $M \in \mathbb{F}_q^{n\times r}$ such that:*

- *each column of M is contained in C;*
- *the i-th row of M is contained in V_i; and*
- *the row-span of M is U.*

We call such an M a witness matrix*, and columns of M* witness columns*.*

In [27], an LCL property is given by a collection $\mathcal{F}$ of r-local profiles $\mathbf{V}$. Our key generalization is to also record the corresponding row-spans in the family. That is, $\mathcal{F}$ is no longer a collection of profiles $\mathbf{V}$; now it is a collection of *pairs* $(\mathbf{V}, U)$, where U constrains the row-span of a witness matrix.

▶ **Definition 21** (Row-span Constrained LCL Property)**.** *A* row-span constrained r-local LCL family $\mathcal{F}$ *is a collection of* r-local pairs $(\mathbf{V}, U)$ *where* $\mathbf{V} = (V_1, \ldots, V_n) \in \mathcal{L}(\mathbb{F}_q^r)^n$ *is an r-local profile and* $U \in \mathcal{L}(\mathbb{F}_q^r)$ *is the row-span constraint of the witness matrix.*

A code C contains *$\mathcal{F}$ if it contains some pair $(\mathbf{V}, U) \in \mathcal{F}$. It* avoids *$\mathcal{F}$ otherwise. We say that the* row-span constrained r-local property *given by $\mathcal{F}$ is the property of* avoiding *$\mathcal{F}$.*[7]

▶ Remark 22. Definition 21 generalizes the definition of an LCL property in [27], in the sense that they restrict the columns of their witness matrices M to be distinct; as they observe, this is the same as restricting the row-span of the witness matrices to lie in $\mathcal{L}_{\text{dist}}(\mathbb{F}_q^r)$, where $\mathcal{L}_{\text{dist}}(\mathbb{F}_q^r)$ denotes all subspaces U of $\mathbb{F}_q^r$ so that no two distinct coordinates $i, j \in [r]$ are identically equal on U.

Next, we define what it means for an $\mathbb{F}_q$-additive code to contain $(\mathbf{V}, U)$.

▶ **Definition 23** (Folded containment)**.** *Let $C \subset (\mathbb{F}_q^s)^n$ be an $\mathbb{F}_q$-additive code. For an r-local profile $\mathbf{V} = (V_1, \ldots, V_n) \in (\mathcal{L}(\mathbb{F}_q))^n$, define the* duplicated profile $\mathbf{V}^{\otimes s} \in (\mathcal{L}(\mathbb{F}_q))^{sn}$ *by*

$$V^{\otimes s}_{(i,j)} = V_i \tag{6}$$

for any $i \in [n]$ and $j \in [s]$. We say that C contains $(\mathbf{V}, U)$ if the "unfolded" version $C \subset \mathbb{F}_q^{sn}$ contains $(\mathbf{V}^{\otimes s}, U)$. Here, by the "unfolded code," we mean to embed $(\mathbb{F}_q^s)^n$ in $\mathbb{F}_q^{sn}$ in the natural way.

For a family $\mathcal{F}$ of pairs, define $\mathcal{F}^{\otimes s} = \{(\mathbf{V}^{\otimes s}, U) : (\mathbf{V}, U) \in \mathcal{F}\}$. Thus the $\mathbb{F}_q$-additive code $C \subset (\mathbb{F}_q^s)^n$ contains $\mathcal{F}$ if and only if the "unfolded" version $C \subset \mathbb{F}_q^{sn}$ contains $\mathcal{F}^{\otimes s}$.

▶ Remark 24. Note that $\mathcal{F}^{\otimes s}$ does not encode the same property as $\mathcal{F}$ applied to unfolded codes. For example, proximity is not preserved: being far as an $\mathbb{F}_q$-additive code over $\Sigma = \mathbb{F}_q^s$ does not mean it is far when we unfold, since we can change one coordinate per block of s. The onus is on the property proving step, to show for any additive code, the desired property profiles take the form (6) for every s.

Now, we define the potential function and the threshold of row-span constrained r-local LCL properties identically as [27], up to slightly modified notational conventions and normalization by n. We emphasize that these definitions are code-independent.

▶ **Definition 25** (Potential and Threshold, [27])**.** *For an r-local pair $(\mathbf{V}, U)$, and a real number R, define the* potential function

$$\phi_{\mathbf{V}}(U, R) \coloneqq (R-1)\dim U + \mathbb{E}_{i \in [n]} \dim(U \cap V_i). \tag{7}$$

[7] We note that other sources, like [27], define an r-local property to be the property of *containing* $\mathcal{F}$. Since we are concerned with the property of avoiding a family $\mathcal{F}$, we define it this way.

For $W \subsetneq U$, define

$$R_{\mathbf{V}}(U, W) \coloneqq 1 - \mathbb{E}_{i\in[n]}\left[\frac{\dim(U \cap V_i) - \dim(W \cap V_i)}{\dim U - \dim W}\right]. \tag{8}$$

Define threshold *of the r-local pair $(\mathbf{V}, U)$ as*

$$R_{\mathbf{V}}(U) \coloneqq \max_{W\subsetneq U} R_{\mathbf{V}}(U, W). \tag{9}$$

For a row-span constrained r-local LCL property, define

$$R_{\mathcal{F}} \coloneqq \min_{(\mathbf{V},U)\in\mathcal{F}} R_{\mathbf{V}}(U). \tag{10}$$

The only difference between the definition above and that in [27] is the fact that the minimum in (10) is over profile-subspace pairs $(\mathbf{V}, U) \in \mathcal{F}$ rather than over profiles $\mathbf{V} \in \mathcal{F}$.

Next, we unpack the definitions to interpret the threshold as a root of the potential function.

▶ **Lemma 26** (Threshold interpretation). *For every r-local profile $\mathbf{V}$, $W \subsetneq U \in \mathcal{L}(\mathbb{F}_q^r)$, and $R \in \mathbb{R}$,*

$$\phi_{\mathbf{V}}(U, R) - \phi_{\mathbf{V}}(W, R) = (\dim U - \dim W)(R - R_{\mathbf{V}}(U, W)). \tag{11}$$

Let $\mathcal{F}$ be a row-span constrained r-local LCL family and let $\eta \geq 0$. We observe that

1. *If $R \leq R_{\mathcal{F}} - \eta$, then for every $(\mathbf{V}, U) \in \mathcal{F}$ there exists $W \subsetneq U$ such that $\phi_{\mathbf{V}}(U, R) - \phi_{\mathbf{V}}(W, R) \leq -\eta$.*
2. *If $R \geq R_{\mathcal{F}} + \eta$, then there exists $(\mathbf{V}, U) \in \mathcal{F}$ such that for every $W \subsetneq U$, $\phi_{\mathbf{V}}(U, R) - \phi_{\mathbf{V}}(W, R) \geq \eta$.*

We also see that by our definition of profile containment for additive (rather than linear) codes, the threshold rate R_V is preserved for profiles taking tensor form in (6) as s changes. Intuitively, this is because $(U, \mathbf{V}, W)$ are all independent of s, so potential function $\phi_{\mathbf{V}^{\otimes s}}(U, R) = \phi_{\mathbf{V}}(U, R)$. Then, $R_{\mathbf{V}^{\otimes s}}(U, W)$ is simply its root. We verify this directly.

▶ **Proposition 27.** *Let $(\mathbf{V}, U)$ be any r-local pair on plain codes in $\mathbb{F}_q^n$ and let $W \subsetneq U$. Then, for every $s \in \mathbb{N}$, $R_{V^{\otimes s}}(U, W) = R_V(U, W)$. Consequently, for any row-span constrained local LCL family, $R_{\mathcal{F}} = R_{\mathcal{F}^{\otimes s}}$ is independent of s.*

3.2 Behavior Under Quotients

We next record how local profile containment behaves under quotient maps. Again, omitted proofs can be found in [18].

▶ Remark 28 (Comparison to [17]). In [17], the spaces V_i, U can lie in an arbitrary vector space over $\mathbb{F}_q$. In contrast, our definition of r-local pairs require $V_i, U \subset \mathbb{F}_q^r$ and is thus more concrete. However, this is at the expense of a less elegant quotient behavior: since our profiles are coordinate-dependent, a quotient by $W \subsetneq U$ is implemented by choosing a surjective linear map $\pi : \mathbb{F}_q^r \to \mathbb{F}_q^{r'}$ where $\ker \pi = W$. This is the same as quotienting by W and choosing coordinates on the quotient.

▶ **Definition 29** (Quotient profile). *Let $\mathbf{V} = (V_1, \ldots, V_n)$ be an r-local pair and let $\pi : \mathbb{F}_q^r \to \mathbb{F}_q^{r'}$ be a surjective linear map, so $r' \leq r$. Define the r'-local profile $\pi\mathbf{V} \coloneqq (\pi V_1, \ldots, \pi V_n)$. For an r-local pair $(\mathbf{V}, U)$ and surjective linear map π, if $W = \ker \pi \subsetneq U \subset \mathbb{F}_q^r$, then we say $(\pi\mathbf{V}, \pi U)$ is a quotient pair of $(\mathbf{V}, U)$. Then, the condition $W \subsetneq U$ is equivalent to $\pi U \neq 0$.*

While the locality is changed in a quotient pair which in particular means it is no longer in $\mathcal{F}$, we will see next that the potential, threshold, and code-containment are well-behaved.

▶ **Lemma 30** (Quotient Containment). *Suppose code C contains r-local pair $(\mathbf{V}, U)$ and $\pi : \mathbb{F}_q^r \to \mathbb{F}_q^{r'}$ is a surjective linear map such that $\pi U \neq 0$. Then, C contains $(\pi\mathbf{V}, \pi U)$.*

We remark that this definition of code containment works for all folding s.

▶ **Lemma 31** (Quotient Potential). *Let $\pi : \mathbb{F}_q^r \to \mathbb{F}_q^{r'}$ be a surjective linear map with kernel $W \subsetneq U \in \mathcal{L}(\mathbb{F}_q^r)$. Then, for any $R \in \mathbb{R}$, $\phi_{\pi\mathbf{V}}(\pi U, R) = \phi_{\mathbf{V}}(U, R) - \phi_{\mathbf{V}}(W, R)$.*

Finally, we present a useful corollary of the low-rate code profile containment via the quotient characterization. We emphasize that this is code-independent, and reduces the low-rate directions of the threshold proofs of various codes into simple code-specific counting.

▶ **Lemma 32** (Maximal Quotient). *If C contains an r-local pair $(\mathbf{V}, U)$ and $R \leq R_{\mathbf{V}}(U) - \eta$ for some $R, \eta \geq 0$, then there exists a surjective linear map $\pi : \mathbb{F}_q^r \to \mathbb{F}_q^{r'}$ such that*

- *$W \coloneqq \ker \pi \subsetneq U$,*
- *C contains r'-local pair $(\pi\mathbf{V}, \pi U)$,*
- *$\phi_{\pi\mathbf{V}}(\pi U, R) \leq -\eta \dim \pi U$.*

Moreover, we can choose π such that $\phi_{\pi\mathbf{V}}(U', R) \leq -\eta \dim U'$ for all nonzero subspaces $U' \subset \pi U$.

3.3 Thresholds for Random Ensembles of Codes

In this subsection, we state fixed-pair and family-level threshold results for the random ensembles considered here; omitted proofs can be found in [18]. The quantity $R_{\mathbf{V}}(U)$ is the fixed-pair threshold, and a union bound over pairs gives the corresponding statement for a row-span constrained LCL family $\mathcal{F}$.

3.3.1 Random Linear Codes

▶ **Lemma 33.** *Let $(\mathbf{V}, U)$ be an r-local pair, let $\mathcal{M}_{\mathbf{V}}(U)$ be the set of matrices $M \in \mathbb{F}_q^{n \times r}$ with row-span U that satisfy $\mathbf{V}$, and let $\mathcal{M}^{\star}_{\mathbf{V}}(U)$ be those with row-span contained in U. For any $R \in \mathbb{R}$, $|\mathcal{M}_{\mathbf{V}}(U)| \leq |\mathcal{M}^{\star}_{\mathbf{V}}(U)| = q^{n((1-R)\cdot \dim U + \phi_{\mathbf{V}}(U,R))}$.*

▶ **Theorem 34** (Fixed-Pair Random Linear Code Threshold). *Let $(\mathbf{V}, U)$ be an r-local pair and let $C \subset \mathbb{F}_q^n$ be a random linear code of rate R.*

1. *If $R \leq R_{\mathbf{V}}(U) - \eta$ for some $\eta > 0$, then $\mathbb{P}(C \text{ contains } (\mathbf{V}, U)) \leq q^{-\eta n}$.*
2. *If $R \geq R_{\mathbf{V}}(U) + \eta$ for some $\eta > 0$, then $\mathbb{P}(C \text{ contains } (\mathbf{V}, U)) \geq 1 - q^{r^2 - \eta n}$.*

The low-rate direction (1) will essentially follow from Lemma 32 and a few simple lemmas that we state first.

▶ **Lemma 35** ([27, Lemma 4.1]). *If $C \subset \mathbb{F}_q^n$ is an RLC of rate R, then for any $M \in \mathbb{F}_q^{n \times r}$*

$$\mathbb{P}(M_{\star j} \subset C \,\forall\, j \in [r]) = q^{(R-1)n \cdot \mathrm{rank}(M)} \tag{12}$$

The idea behind Lemma 35 is to view C as the kernel of $(1-R)n$ many linear constraints in $\mathbb{F}_q^n$ which column-span of M needs to satisfy. Each of $\mathrm{rank}(M)$ basis vectors satisfies them with probability $q^{-(1-R)n}$. We note that this is the only RLC-specific ingredient we use.

Combining with Lemma 33, we obtain the following corollary, which intuitively explains why $\phi_{\mathbf{V}}(U, R)$ characterizes the containment of $(\mathbf{V}, U)$ in a random linear code of rate C.

▶ **Corollary 36.** *The expected number of witnesses matrices M for the containment of an r-local pair $(\mathbf{V}, U)$ in a random linear code C with rate $R \in [0,1]$ is at most $q^{n\phi_{\mathbf{V}}(U,R)}$.*

With these lemmas, we deduce Theorem 34.

Proof of Theorem 34. For the low-rate case (1), if C contains $(\mathbf{V}, U)$, then by Lemma 32, there is a surjective linear map $\pi = \pi_{(\mathbf{V},U)} : \mathbb{F}_q^r \to \mathbb{F}_q^{r'}$ such that C contains $(\pi\mathbf{V}, \pi U)$ and $\phi_{\pi\mathbf{V}}(\pi U, R) \le -\eta$. By a union bound and Corollary 36,

$$\mathbb{P}(C \text{ contains } (\mathbf{V}, U)) \le \mathbb{P}(C \text{ contains } (\pi\mathbf{V}, \pi U)) \le q^{n\phi_{\pi\mathbf{V}}(\pi U, R)} \le q^{-\eta n}. \tag{13}$$

For the high-rate case (2), we know that $R \ge R_{\mathbf{V}}(U, W) + \eta$ for every $W \subsetneq U$. Then, $\phi_{\mathbf{V}}(U, R) \ge \phi_{\mathbf{V}}(W, R) + \eta$ for all $W \subsetneq U$. Let random variable X_W be the number of witnesses matrices M for the containment by C of an r-local pair $(\mathbf{V}, W)$. By Corollary 36, we know that

$$\mathbb{E}[X_W] \le q^{n\phi_{\mathbf{V}}(W,R)} \tag{14}$$

Now, let us fix any code C of rate $R \in [0,1]$ and consider matrices $\mathcal{M}^{\star}_{\mathbf{V}}(U)$ that satisfies $\mathbf{V}$ and has row-span contained in U. By Lemma 33, we know that

$$|\mathcal{M}^{\star}_{\mathbf{V}}(U)| = q^{n((1-R)\cdot \dim U + \phi_{\mathbf{V}}(U,R))}. \tag{15}$$

Now, for any realization of C, requiring each column of M to lie in C imposes at most $(1-R)n \dim U$ independent constraints, since C has codimension $(1-R)n$ and the row-span is contained in U. Therefore, the number of $M \in \mathcal{M}^{\star}_{\mathbf{V}}(U)$ contained in C is

$$\sum_{W \in \mathcal{L}(U)} X_W \ge |\mathcal{M}^{\star}_{\mathbf{V}}(U)| q^{(R-1)n \dim U} = q^{n\phi_{\mathbf{V}}(U,R)} \tag{16}$$

deterministically for every realization C. Now, C contains $(\mathbf{V}, U)$ if and only if $X_U > 0$, so

$$\begin{aligned}
\mathbb{P}(C \text{ does not contain } (\mathbf{V}, U)) &= \mathbb{P}(X_U = 0) \\
&\le \mathbb{P}\left(\sum_{W \subsetneq U} X_W \ge q^{n\phi_{\mathbf{V}}(U,R)} \right) \\
&\le \frac{\sum_{W \subsetneq U} \mathbb{E}[X_W]}{q^{n\phi_{\mathbf{V}}(U,R)}} \\
&\le \sum_{W \subsetneq U} q^{n\phi_{\mathbf{V}}(W,R) - n\phi_{\mathbf{V}}(U,R)} \\
&\le |\mathcal{L}(U)| q^{-\eta n}
\end{aligned} \tag{17}$$

where we apply Markov's inequality and (14). Now, the conclusion follows $|\mathcal{L}(U)| \le q^{r^2}$. ◀

We now lift Theorem 34 to the whole row-span constrained LCL family $\mathcal{F}$.

▶ **Corollary 37** (Random Linear Code Threshold)**.** *Let $\mathcal{F}$ be a row-span constrained r-local property and let $C \subset \mathbb{F}_q^n$ be a random linear code of rate R.*

1. *If $R \le R_{\mathcal{F}} - \eta$ for some $\eta > 0$, then $\mathbb{P}(C \text{ contains } \mathcal{F}) \le |\mathcal{F}| q^{-\eta n}$.*
2. *If $R \ge R_{\mathcal{F}} + \eta$ for some $\eta > 0$, then $\mathbb{P}(C \text{ contains } \mathcal{F}) \ge 1 - q^{r^2 - \eta n}$.*

Therefore, when $|\mathcal{F}| = q^{o(n)}$[8], we know that $R_{\mathcal{F}}$ is the threshold for RLCs for large n and q: RLCs of rate less than $R_{\mathcal{F}}$ avoid $\mathcal{F}$ with high probability; RLCs of rate greater than $R_{\mathcal{F}}$ contain $\mathcal{F}$ with high probability.

[8] Some previous literature, e.g. [27], call such $\mathcal{F}$ *reasonable*

3.3.2 Random Reed-Solomon Codes

We next record the analogue of the low-rate local threshold Corollary 37(1) for random Reed-Solomon code: if the rate is below the threshold of a row-span constrained LCL family, the random Reed-Solomon code is unlikely to contain the family. We use the following result from [27] as a black box.

▶ **Lemma 38** ([27, Proposition 6.1]). *Let $n \le q$ with q a prime power, and let $r \in \mathbb{N}$. Let $\mathbf{V} = (V_1, \cdots, V_n)$ be an r-local profile, $U \in \mathcal{L}(\mathbb{F}_q^r)$, and let $0 \le R \le 1$ such that for all $U' \in \mathcal{L}(U)$, $\phi_{\mathbf{V}}(U', R) \le -\varepsilon \cdot \dim U'$. Then, with probability taken over uniformly random $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$, we have*

$$\mathbb{P}\big(C = \mathsf{RS}_{\mathbb{F}_q}(\alpha_1, \ldots, \alpha_n, k) \text{ contains } V \text{ with span } U \neq \{0\}\big) \le (2^r - 1)\left(\frac{(4r)^{4r} k}{\varepsilon q}\right)^{\varepsilon n / 2r}.$$

▶ **Corollary 39** (Random Reed-Solomon Code Threshold: Low Rate Direction). *Let $\mathcal{F}$ be a row-span constrained r-local property and let $C \subset \mathbb{F}_q^n$ be a random Reed-Solomon code of rate R. If $R \le R_{\mathcal{F}} - \eta$ for some $\eta > 0$, then*

$$\mathbb{P}(C \text{ contains } \mathcal{F}) \le |\mathcal{F}|(2^r - 1)\left(\frac{(4r)^{4r} R n}{\eta q}\right)^{\eta n / 2r} \tag{18}$$

3.3.3 Random LDPCs (Gallager's Ensemble)

We next record the analogue of the low-rate local threshold Corollary 37(1) and Corollary 39 for random LDPC codes drawn from Gallager's Ensemble (Definition 7).

There is one important difference from random linear codes. For RLCs, the probability that a fixed matrix M is contained in the code is exactly $q^{(R-1)n \operatorname{rank}(M)}$ by Lemma 35. For LDPC codes C, this exact identity is false. Instead, [28] proves that, after conditioning on the usual good-distance event defined for $C = \mathsf{RLDPC}(n, q, s, R)$ as

$$\mathsf{Good} := \left\{\Delta(C) \ge \frac{1}{2} h_q^{-1}(1 - R)\right\}, \tag{19}$$

then C contains any fixed matrix with essentially the same probability as a random linear code, up to a factor $q^{\varepsilon n}$. This is the only LDPC-specific input we need.

We first recall the lemma from [28] saying that Good happens with high probability.

▶ **Lemma 40** ([28, Theorem 2.14]). *For any finite field $\mathbb{F}_q$ and $R \in [0, 1 - 1/q)$, there exists $s_0 = s_0(R, q)$ such that for every $s > s_0$, with probability over $C = \mathsf{RLDPC}(n, q, s, R)$, we have $\mathbb{P}(\mathsf{Good}) := \mathbb{P}\big(\Delta(C) \ge \frac{1}{2} h_q^{-1}(1 - R)\big) \ge 1 - o_{n\to\infty}(1)$.*

The next lemma is the fixed-pair containment estimate for row-span constrained local profiles. This lemma follows [28]; we restate it in our framework and refer to the full version [18] for the proof.

▶ **Lemma 41** (RLDPCs containing local profiles). *For any $\varepsilon, R \in (0, 1)$, positive integer r, finite field $\mathbb{F}_q$, there exists an $s_0 = s_0(\varepsilon, r, R, q)$ such that the following holds for all odd $s > s_0$ and sufficiently large n. Let $\mathbf{V} = (V_1, \ldots, V_n)$ be an r-local profile and let $U \in \mathcal{L}(\mathbb{F}_q^r)$. For $C = \mathsf{RLDPC}(n, q, s, R)$, $\mathbb{P}(C \text{ contains } (\mathbf{V}, U) | \mathsf{Good}) \le q^{n\phi_{\mathbf{V}}(U,R) + \varepsilon n}$, where $\mathsf{Good} := \big\{\Delta(C) \ge \frac{1}{2} h_q^{-1}(1 - R)\big\}$.*

The LDPC threshold statement differs from its RLC analogue only by conditioning on the good-distance event Good and by the additional $q^{\eta n/2}$ slack from Lemma 41. The complete derivation appears in the full version [18].

▶ **Theorem 42** (Fixed-Pair Random LDPC Threshold). *Let $(\mathbf{V}, U)$ be an r-local pair and let $C = \mathsf{RLDPC}(n, q, s, R)$ be a random LDPC code from Gallager's ensemble. Define the event* $\mathsf{Good} := \left\{\Delta(C) \geq \frac{1}{2} h_q^{-1}(1-R)\right\}$.

For every $R, \eta \in (0,1)$, positive integer r, and finite field $\mathbb{F}_q$, there exists $s_0 = s_0(\eta, r, R, q)$ such that for every odd $s > s_0$, the following holds for all sufficiently large n:

1. *If $R \leq R_{\mathbf{V}}(U) - \eta$, then $\mathbb{P}(C \text{ contains } (\mathbf{V}, U) | \mathsf{Good}) \leq q^{-\eta n/2}$, and so*

$$\mathbb{P}(C \text{ contains } (\mathbf{V}, U)) \leq q^{-\eta n/2} + o_{n\to\infty}(1). \tag{20}$$

2. *If $R \geq R_{\mathbf{V}}(U) + \eta$, then $\mathbb{P}(C \text{ contains } (\mathbf{V}, U) | \mathsf{Good}) \geq 1 - q^{r^2 - \eta n/2}$, and so*

$$\mathbb{P}(C \text{ contains } (\mathbf{V}, U)) \geq 1 - q^{r^2 - \eta n/2} - o_{n\to\infty}(1). \tag{21}$$

▶ Remark 43. Only part (1) of Theorem 42 is needed later; part (2) is recorded for completeness, and the proofs of both parts appear in the full version [18].

This should not be confused with a converse to the local-property transfer theorem for RLDPC codes ([28, Theorem 1.9]), which is false at that level of generality, as observed in [28]; our argument uses the additional linear coordinate-wise profile structure of row-span constrained LCL properties.

As before, the corresponding family-level statement is the following.

▶ **Corollary 44** (Random LDPC Code Threshold). *Let $\mathcal{F}$ be a row-span constrained r-local property and let $C = \mathsf{RLDPC}(n, q, s, R)$ be a random LDPC code from Gallager's ensemble (Definition 7). For every $R, \eta \in (0,1)$, positive integer r, and finite field $\mathbb{F}_q$, there exists $s_0 = s_0(\eta, r, R, q)$ such that for every odd $s > s_0$, the following holds for all sufficiently large n:*

1. *If $R \leq R_{\mathcal{F}} - \eta$, then $\mathbb{P}(C \text{ contains } \mathcal{F}) \leq |\mathcal{F}| q^{-\eta n/2} + o_{n\to\infty}(1)$*
2. *If $R \geq R_{\mathcal{F}} + \eta$, then $\mathbb{P}(C \text{ contains } \mathcal{F}) \geq 1 - q^{r^2 - \eta n/2} - o_{n\to\infty}(1)$*

3.4 Threshold for Subspace Design Codes

Informally, for ordinary r-local properties where every row-span $U \in \mathcal{L}_{\text{dist}}(\mathbb{F}_q^r)$ is permitted for each profile $\mathbf{V}$, [9, Theorems 3.1, 4.2] shows that the threshold of containing $\mathbf{V}$ is dictated by $\tau(r)$ versus $R_{\mathbf{V}}$. For good subspace design codes, like folded Reed-Solomon codes, we have $\tau(r) \approx R$, which means that good subspace design codes and random linear codes share the same threshold.

We generalize these results to row-span constrained local properties. We emphasize that the transfer works at the level of any pair $(\mathbf{V}, U)$ via the following theorem.

▶ **Theorem 45** (Fixed-Pair Subspace Design Code Threshold). *Let $(\mathbf{V}, U)$ be an r-local pair.*

1. *Let $C \subset (\mathbb{F}_q^s)^n$ be an additive τ-subspace design code. If $\tau(r) < R_{\mathbf{V}}(U)$, then C does not contain $(\mathbf{V}, U)$.*
2. *For any $\eta > 0$, there exists a τ-subspace design additive code $C \subset (\mathbb{F}_q^s)^n$ with $\tau(r) \leq R_{\mathbf{V}}(U) + \eta$ that contains $(\mathbf{V}, U)$.*

The statement (1) in Theorem 45 essentially follows from Lemma 32 and the subspace design condition.

Proof of Theorem 45(1). Suppose $\tau(r) \leq R_{\mathbf{V}}(U) - \eta$, then by Lemma 32 there exists surjective linear map $\pi : \mathbb{F}_q^r \to \mathbb{F}_q^{r'}$ such that $\phi_{\pi\mathbf{V}}(\pi U, \tau(r)) < 0$ and C contains $(\pi\mathbf{V}, \pi U)$. The latter implies the existence of a witness matrix M. Let the columns of the witness matrix M for $(\pi\mathbf{V}, \pi U)$ be $c_j = \text{Enc}(a_j)$ and define the message subspace $A := \text{span}(a_j)$. Since the encoder is injective and the witness matrix has row-span πU, $d := \dim A = \dim(\pi U)$.

For each coordinate $i \in [n]$, the image $\mathrm{Enc}_i(A) \subset \mathbb{F}_q^s$ is the column span of the i-th "block" of the witness matrix. The rows of that block lie in $\pi U \cap \pi V_i$, hence

$$\dim(\pi U \cap \pi V_i) \geq \dim \mathrm{Enc}_i(A) = \dim A - \dim(A \cap \ker(\mathrm{Enc}_i)). \tag{22}$$

where the equality follows from rank-nullity. Averaging over i and applying the subspace design condition Definition 15 with $d = \dim A \leq r$, we obtain that

$$d - \mathbb{E}_{i\in[n]} \dim(\pi U \cap \pi V_i) \leq \mathbb{E}_{i\in[n]} \dim(A \cap \ker(\mathrm{Enc}_i)) \leq \tau(r)d. \tag{23}$$

Rearranging gives $\phi_{\pi\mathbf{V}}(\pi U, \tau(r)) \geq 0$, a contradiction. Thus, C does not contain $(\mathbf{V}, U)$. ◀

For the statement (2) in Theorem 45, we use the following theorem from [9] that shows RLCs are good subspace design codes when viewed as an $\mathbb{F}_q$-additive code with alphabet $\mathbb{F}_q^s$.

▶ **Theorem 46** ([9, Theorem 4.1]). *Fix any $\eta > 0$. A random $\mathbb{F}_q$-additive code $C \subset (\mathbb{F}_q^s)^n$ with rate $R = k/sn$ satisfies*

$$\mathbb{P}(\tau(r) \leq R + \eta) \geq 1 - rq^{-\eta srn + 3(r^2+1)n}. \tag{24}$$

Proof of Theorem 45(2). Fix $\eta > 0$ and let $R = R_{\mathbf{V}}(U) + \eta/2$. For sufficiently large s, we know by Theorem 46 that, for a random $\mathbb{F}_q$-additive code $C \subset (\mathbb{F}_q^s)^n$ of rate R,

$$\mathbb{P}\Big(\tau(r) \leq R_{\mathbf{V}}(U) + \eta = R + \frac{\eta}{2}\Big) \geq 1 - o_{s\to\infty}(1), \tag{25}$$

On the other hand, $R = R_{\mathbf{V}}(U) + \eta/2 = R_{\mathbf{V}^{\otimes s}}(U) + \eta/2$ by Proposition 27. By Theorem 34,

$$\mathbb{P}(C \text{ contains } (\mathbf{V}, U)) = \mathbb{P}(\text{unfolded } C \text{ contains } (\mathbf{V}^{\otimes s}, U)) \geq 1 - q^{r^2 - \eta sn/2} = 1 - o_{s\to\infty}(1). \tag{26}$$

Therefore, we can pick sufficiently large s such that with probability $2/3$, C contains $(\mathbf{V}, U)$ and is a τ-subspace design code with $\tau(r) \leq R_{\mathbf{V}}(U) + \eta$. This proves (2). ◀

▶ Remark 47. Compared to [9, Theorem 4.2], we take the subspace design parameter d in $\tau(d)$ to be $d = r$ for simplicity.

We can obtain analogous threshold statements for $R_{\mathcal{F}}$ by taking the minimum over $\mathcal{F}$.

▶ **Corollary 48** (Subspace Design Code Threshold). *Let $\mathcal{F}$ be any row-span constrained r-local LCL family. Then*

1. *Let $C \subset (\mathbb{F}_q^s)^n$ be an additive τ-subspace design code. If $\tau(r) < R_{\mathcal{F}}$, then C avoids $\mathcal{F}$.*
2. *For any $\eta > 0$, there exists a τ-subspace design additive code $C \subset (\mathbb{F}_q^s)^n$ with $\tau(r) \leq R_{\mathcal{F}} + \eta$ that contains $\mathcal{F}$.*

4 Curve-Decoding as a Row-Span Constrained LCL Property

Recall the definition of curve-decoding:

▶ **Definition 10** (Curve-Decodability). *An additive code $C \subset \Sigma^n$ is (ℓ, δ, a, b) curve-decodable if for every $u_0, u_1, \cdots, u_\ell \in \Sigma^n$, all functions $f : \mathbb{F}_q \to C$, whenever the set $A = \Big\{\alpha \in \mathbb{F} : \Delta\big(\sum_{j=0}^{\ell} u_j \alpha^j, f(\alpha)\big) \leq \delta\Big\}$ has at least a elements, there exist $c_0, c_1, \cdots, c_\ell \in C$ such that $\Big|\Big\{\alpha \in A : f(\alpha) = \sum_{j=0}^{\ell} c_j \alpha^j\Big\}\Big| \geq b$.*

As observed in [16] and discussed in Section 1.2, curve-decoding does not seem to fit immediately into the LCL framework. This section states its exact casting as a *row-span constrained* LCL property and refers to the full version [18] for the proof. Before we describe the correct row-span constraint, we define the degree-ℓ Reed-Solomon evaluation subspace on B by

$$\mathrm{RS}_\ell(B) \coloneqq \{(p(\alpha))_{\alpha\in B} : p \in \mathbb{F}_q[x],\ \deg p \le \ell\} \subset \mathbb{F}_q^B.$$

Thus $\mathrm{RS}_\ell(B)$ is the space of scalar degree-ℓ evaluation patterns on the parameter set B.

▶ **Definition 49** (Curve-free Subspaces). *Let $A \subset \mathbb{F}_q$. A subspace $U \in \mathcal{L}(\mathbb{F}_q^A)$ is (ℓ, A, b)* curve-free *if $\pi_B(U) \not\subset \mathrm{RS}_\ell(B)$ for every $B \in \binom{A}{b}$, where $\pi_B : \mathbb{F}_q^A \to \mathbb{F}_q^B$ denote coordinate restriction to B.*

For this definition to be non-trivial, we assume $\ell + 1 \le b \le |A|$. Now, we observe the following lemma, which translates (ℓ, A, b) curve-freeness of the row-span of a matrix to the condition that no b columns lie on a degree-ℓ curve; see [18] for the proof.

▶ **Lemma 50.** *Let $B \subset A \subset \mathbb{F}_q$ and let N be a positive integer.*[9] *Fix a matrix $M \in \mathbb{F}_q^{N\times A}$ with row-span $U \in \mathcal{L}(\mathbb{F}_q^A)$. Then, $\pi_B(U) \subset \mathrm{RS}_\ell(B)$ if and only if there exists a degree-ℓ curve $g : \mathbb{F}_q \to \mathbb{F}_q^N$ such that $g(\alpha) = M_{\star\alpha}$ for every $\alpha \in B$.*

Observe that latter condition is exactly the kind of codeword condition we wish to impose on columns of witness matrices for the failure of (ℓ, δ, a, b) curve-decoding. Lemma 50 gives an equivalent row-span constraint. We make this more formal below.

▶ **Corollary 51.** *Let $A \subset \mathbb{F}_q$ and let $M \in \mathbb{F}_q^{N\times A}$ be a matrix with row-span U whose columns lie in a linear code $C \subset \mathbb{F}_q^N$. If $\ell < b \le |A|$, then U is (ℓ, A, b) curve-free if and only if no b columns of M lie on a degree-ℓ "codeword curve." (That is, if and only if there is no curve $c(x) = \sum_{i=0}^{\ell} c_i x^i$ for $c_i \in C$ and no set $B \in \binom{A}{b}$ so that $c(\alpha) = M_{\star\alpha}$ for all $\alpha \in B$.)*

We now cast curve-decoding as a row-span constrained LCL property, thereby bypassing the notion of $\bigvee$-decodability from [16], and the parameter losses that arose from it.

▶ **Theorem 52** (Locality of curve-decoding). *Let $\ell, a, b \in \mathbb{N}$ with $\ell + 1 \le b \le a \le q$ and let $\delta \in (0, 1)$. Then there exists a row-span constrained a-LCL family $\mathcal{F}$ with $|\mathcal{F}| \le q^{a+a^2} 2^{an}$ such that every $\mathbb{F}_q$-additive code C is not (ℓ, δ, a, b) curve-decodable if and only if C contains $\mathcal{F}$.*

Proof. Fix indexing sets $A \in \binom{\mathbb{F}_q}{a}$ and $I_\alpha \subset [n]$ with $|I_\alpha| \ge (1-\delta)n$ for each $\alpha \in A$. For each choice of $A, \{I_\alpha\}_{\alpha\in A}$, we will define an r-local profile $\mathbf{V} = \mathbf{V}(A, \{I_\alpha\}_{\alpha\in A}) = (V_1, \ldots, V_n)$ as follows. For each $i \in [n]$, let $A_i \coloneqq \{\alpha \in A : i \in I_\alpha\}$. Define

$$V_i \coloneqq \left\{y \in \mathbb{F}_q^A : (y_\alpha)_{\alpha\in A_i} \in \mathrm{RS}_\ell(A_i)\right\}. \tag{27}$$

Equivalently, $y \in V_i$ if there exists a polynomial $p \in \mathbb{F}_q[x]$ of degree at most ℓ such that $p(\alpha) = y_\alpha$ for every $\alpha \in A_i$. This is clearly a linear subspace of $\mathbb{F}_q^A$. Now define

$$\mathcal{F} \coloneqq \left\{(\mathbf{V}(A, \{I_\alpha\}_{\alpha\in A}), U) : A \in \binom{\mathbb{F}_q}{a},\ I_\alpha \in \binom{[n]}{\ge (1-\delta)n},\ U \in \mathcal{L}(\mathbb{F}_q^A) \text{ is } (\ell, A, b) \text{ curve-free}\right\}. \tag{28}$$

We show the bound on the size of $\mathcal{F}$: There are $\binom{q}{a} \le q^a$ choices for A, at most 2^n choices for each $I_\alpha \subset [n]$, and the number of choices of U is at most $|\mathcal{L}(\mathbb{F}_q^a)| \le q^{a^2}$.

Next, we prove that not being curve-decodable is equivalent to containing $\mathcal{F}$.

[9] Think of N as either the block length $N = n$ of a linear code $C \subset \mathbb{F}_q^n$; or the block length $N = sn$ of an "unfolded" $\mathbb{F}_q$-additive code $C \subset (\mathbb{F}_q^s)^n$, thought of as $C \subset \mathbb{F}_q^{sn}$.

Not curve-decodable ⇒ Contains $\mathcal{F}$. If $C \subset (\mathbb{F}_q^s)^n$ is not (ℓ, δ, a, b) curve-decodable, then there exists $A \in \binom{\mathbb{F}_q}{a}$, a degree-$\ell$ curve $u : \mathbb{F}_q \to (\mathbb{F}_q^s)^n$, and codewords $f(\alpha) \in C$ such that $\Delta(u(\alpha), f(\alpha)) \leq \delta$ for every $\alpha \in A$, but no degree-ℓ codeword curve passes through $f(\alpha)$ for b-many of α. For $\alpha \in A$, define the agreement set $I_\alpha \coloneqq \{i \in [n] : u(\alpha)_i = f(\alpha)_i\}$, so $|I_\alpha| \geq (1-\delta)n$. Form the matrix $M \in \mathbb{F}_q^{sn \times A}$ with columns $M_{\star\alpha} = f(\alpha) \in C$. Index the rows by $[n] \times [s]$ and let U be the row-span of M.

We check the local profile constraints. For any $(i, j) \in [n] \times [s]$, the corresponding row $M_{ij,\star} = (f(\alpha)_{ij})_{\alpha \in A}$. Since $f(\alpha)_{ij} = u(\alpha)_{ij}$ whenever $i \in I_\alpha$, we observe that $M_{ij,\star}$ agrees with a polynomial $x \mapsto u(x)_{ij}$ of degree at most ℓ on A_i, so $M_{ij,\star} \in V_i$ by the definition of V_i in (27).

By assumption, no b of the columns $f(\alpha)$ lie on a degree-ℓ codeword curve. By Corollary 51, the row-span U is (A, ℓ, b) curve-free. Hence M witnesses that C contains $(\mathbf{V}(A, \{I_\alpha\}_{\alpha \in A}), U) \in \mathcal{F}$.

Contains $\mathcal{F}$ ⇒ Not curve-decodable. Suppose C contains the a-local pair $(\mathbf{V}(A, \{I_\alpha\}_{\alpha \in A}), U)$ via witness matrix $M \in \mathbb{F}_q^{sn \times A}$. Let the columns of M be $f(\alpha) \in C$ for $\alpha \in A$.

We construct a degree-ℓ curve close to these codewords. For $(i, j) \in [n] \times [s]$ the row $M_{ij,\star}$ lies in V_i, so by definition (27) of V_i, there exists a polynomial u_{ij} of degree at most ℓ such that $u_{ij}(\alpha) = f(\alpha)_{ij}$ for every $\alpha \in A_i \coloneqq \{\alpha \in A : i \in I_\alpha\}$. Collecting these polynomials over all (i, j) defines a degree-ℓ curve $u : \mathbb{F}_q \to (\mathbb{F}_q^s)^n$. For each $\alpha \in A$, if $i \in I_\alpha$ so $\alpha \in A_i$, then $u_{ij}(\alpha) = f(\alpha)_{ij}$ for every $j \in [s]$. Hence, $u(\alpha)_i = f(\alpha)_i$ for all $i \in I_\alpha$. For every $\alpha \in A$, $|I_\alpha| \geq (1-\delta)n$, so $\Delta(u(\alpha), f(\alpha)) \leq \delta$.

Finally, since U is (A, ℓ, b) curve-free, Corollary 51 implies that no b of the columns $f(\alpha)$ lie on a degree-ℓ codeword curve. Together, u and $f(\alpha)$ witness that C is not (ℓ, δ, a, b) curve-decodable. ◀

We can plug in specific choices of parameters from Theorem 18 for which subspace design codes are curve-decodable, thereby obtaining a lower bound on the threshold rate via Corollary 48.

▶ **Corollary 53.** *Let $\delta, \eta \in (0, 1)$ and integers $\ell, a, b \in \mathbb{N}$ such that*

$$d \coloneqq \left\lceil \frac{\ell + 1}{\eta} \right\rceil \leq a \quad \text{and} \quad b \leq \frac{\eta a}{d + \eta}. \tag{29}$$

Let $\mathcal{F}$ be the row-span constrained a-local LCL family from Theorem 52, so avoiding $\mathcal{F}$ is equivalent to being (ℓ, δ, a, b) curve-decodable. Then, $R_{\mathcal{F}} \geq 1 - \delta - \eta$. Additionally, $|\mathcal{F}| \leq q^{a+a^2} 2^{an}$.

We conclude with a specialization for the common choice of curve-decoding parameters.

▶ **Corollary 54.** *Let $\ell \in \mathbb{N}$ and $\delta, \eta \in (0, 1)$. Define $a, b, d \in \mathbb{N}$ by*

$$d \coloneqq \left\lceil \frac{\ell + 1}{\eta} \right\rceil, \qquad b \coloneqq \left\lceil \frac{4\ell}{\eta} + \ell \right\rceil, \qquad a \coloneqq \left\lceil \frac{100\ell^2}{\eta^3} \right\rceil. \tag{30}$$

Let $\mathcal{F}$ be the row-span constrained a-local LCL family from Theorem 52, so avoiding $\mathcal{F}$ is equivalent to being (ℓ, δ, a, b) curve-decodable. Then $R_{\mathcal{F}} \geq 1 - \delta - \eta$ and $|\mathcal{F}| \leq q^{a+a^2} 2^{an}$.

5 Improved Proximity Gaps

Finally, we establish proximity gaps (as well as CA and MCA) for three ensembles of codes. For each ensemble, we use Corollary 54 to prove curve-decodability, then we combine this with list-decodability via Corollary 14. We record the final statements below, and refer the reader to [18] for proofs. We begin with a statement for random linear codes.

▶ **Theorem 55** (PG, CA, and MCA for Random Linear Codes). *Let $\ell \in \mathbb{N}$ and $\eta, \delta \in (0,1)$. Let $C \subset \mathbb{F}_q^n$ be a random linear code of rate $R \le 1 - \delta - 2\eta$. If $q \ge \exp(\Omega(\ell^2/\eta^4))$ and $n \ge \Omega(\ell^4/\eta^7)$, then with high probability:*

1. *For every $T \ge 1$, C is curve-decodable with parameters $\left(\ell, \delta, (T-1)\left\lceil \frac{1-R}{\eta} \right\rceil + O\left(\frac{\ell^2}{\eta^3}\right), T\right)$.*
2. *For every $m > 1$, C has correlated agreement, and hence proximity gap, with parameters $\left(\ell, \delta, \frac{\ell m \lceil (1-R)/\eta \rceil + O(\ell^2/\eta^3)}{q}, \frac{1}{m}\right)$.*
3. *C has mutual correlated agreement with parameters $\left(\ell, \delta, \frac{\ell n \lceil (1-R)/\eta \rceil + O(\ell^2/\eta^3)}{q}\right)$.*

Next, we state our main result for Random Reed-Solomon Codes.

▶ **Theorem 56** (PG, CA, and MCA for Random Reed-Solomon Codes). *Let $\ell \in \mathbb{N}$ and $\eta, \delta \in (0,1)$. Let $C \subset \mathbb{F}_q^n$ be a random Reed-Solomon code of rate $R \le 1 - \delta - 2\eta$. If $q \ge n \cdot \exp(\Omega(\ell^4/\eta^7))$ and $n \ge \Omega(\ell^4/\eta^7)$, then with high probability:*

1. *For every $T \ge 1$, C is curve-decodable with parameters $\left(\ell, \delta, (T-1)\left\lceil \frac{1-R}{\eta} \right\rceil + O\left(\frac{\ell^2}{\eta^3}\right), T\right)$.*
2. *For every $m > 1$, C has correlated agreement, and hence proximity gap, with parameters $\left(\ell, \delta, \frac{\ell m \lceil (1-R)/\eta \rceil + O(\ell^2/\eta^3)}{q}, \frac{1}{m}\right)$.*
3. *C has mutual correlated agreement with parameters $\left(\ell, \delta, \frac{\ell n \lceil (1-R)/\eta \rceil + O(\ell^2/\eta^3)}{q}\right)$.*

Finally, we state our result for random LDPC codes.

▶ **Theorem 57** (PG, CA, and MCA for Random LDPC Codes). *Let $\ell \in \mathbb{N}$ and $\eta, \delta \in (0,1)$. There exists $s_0 = s_0(\ell, \eta, \delta, q)$ such that the following holds for every odd $s > s_0$. Let $C = \mathsf{RLDPC}(n, q, s, R)$ be a random LDPC code from Gallager's ensemble with rate $R \le 1 - \delta - 2\eta$. If $q \ge \exp(\Omega(\ell^2/\eta^4))$ and n is sufficiently large, then with high probability:*

1. *For every $T \ge 1$, C is curve-decodable with parameters $\left(\ell, \delta, (T-1)\left\lceil \frac{2(1-R)}{\eta} \right\rceil + O\left(\frac{\ell^2}{\eta^3}\right), T\right)$.*
2. *For every $m > 1$, C has correlated agreement, and hence proximity gap, with parameters $\left(\ell, \delta, \frac{\ell m \lceil 2(1-R)/\eta \rceil + O(\ell^2/\eta^3)}{q}, \frac{1}{m}\right)$.*
3. *C has mutual correlated agreement with parameters $\left(\ell, \delta, \frac{\ell n \lceil 2(1-R)/\eta \rceil + O(\ell^2/\eta^3)}{q}\right)$.*

References

1 Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. In *Proceedings of the 2017 acm sigsac conference on computer and communications security*, pages 2087–2104, 2017. doi:10.1145/3133956.3134104.

2 Gal Arnon, Dan Boneh, and Giacomo Fenzi. Open problems in list decoding and correlated agreement. Cryptology ePrint Archive, Paper 2026/680, 2026. URL: https://eprint.iacr.org/2026/680.

3 Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. WHIR: Reed-Solomon proximity testing with super-fast verification. In *Advances in Cryptology–EUROCRYPT 2025-44th Annual International Conference on the Theory and Applications of Cryptographic Techniques, 2025, Proceedings*, pages 214–243. Springer Science and Business Media Deutschland GmbH, 2025. doi:10.1007/978-3-031-91134-7_8.

4 Vikrant Ashvinkumar, Mursalin Habib, and Shashank Srivastava. Algorithmic improvements to list decoding of folded reed-solomon codes. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 880–898. SIAM, 2026. doi:10.1137/1.9781611978971.35.

5 Eli Ben-Sasson, Dan Carmon, Ulrich Haböck, Swastik Kopparty, and Shubhangi Saraf. On proximity gaps for reed–solomon codes. Cryptology ePrint Archive, Paper 2025/2055, 2025. URL: https://eprint.iacr.org/2025/2055.

6 Eli Ben-Sasson, Dan Carmon, Yuval Ishai, Swastik Kopparty, and Shubhangi Saraf. Proximity gaps for reed–solomon codes. *Journal of the ACM*, 70(5):1–57, 2023. doi:10.1145/3614423.

7 Eli Ben-Sasson, Lior Goldberg, Swastik Kopparty, and Shubhangi Saraf. DEEP-FRI: Sampling Outside the Box Improves Soundness. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 5:1–5:32. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ITCS.2020.5.

8 Eli Ben-Sasson, Swastik Kopparty, and Shubhangi Saraf. Worst-Case to Average Case Reductions for the Distance to a Code. In *33rd Computational Complexity Conference (CCC 2018)*, volume 102 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.CCC.2018.24.

9 Joshua Brakensiek, Yeyuan Chen, Manik Dhar, and Zihan Zhang. From random to explicit via subspace designs with applications to local properties and matroids. *arXiv preprint arXiv:2510.13777*, 2025. doi:10.48550/arXiv.2510.13777.

10 Martijn Brehm, Binyi Chen, Ben Fisch, Nicolas Resch, Ron D Rothblum, and Hadas Zeilberger. Blaze: fast snarks from interleaved raa codes. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 123–152. Springer, 2025. doi:10.1007/978-3-031-91134-7_5.

11 Yeyuan Chen and Zihan Zhang. Explicit folded reed-solomon and multiplicity codes achieve relaxed generalized singleton bounds. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 1–12, 2025. doi:10.1145/3717823.3718114.

12 Elizabeth Crites and Alistair Stewart. On Reed–Solomon proximity gaps conjectures. *Cryptology ePrint Archive*, 2025.

13 Benjamin E Diamond and Angus Gruen. On the distribution of the distances of random words. *Cryptology ePrint Archive*, 2025.

14 Ethereum Foundation. The proximity prize: Reed–solomon challenge, 2026. Accessed: 2026-04-29. URL: https://proximityprize.org/.

15 Robert G Gallager. Low-density parity-check codes. *Low Density Parity Check Codes*, 1963.

16 Rohan Goyal and Venkatesan Guruswami. Optimal proximity gaps for subspace-design codes and (random) reed-solomon codes. *Cryptology ePrint Archive*, 2025.

17 Rohan Goyal, Venkatesan Guruswami, and Jun-Ting Hsieh. Explicit constant-alphabet subspace design codes. *arXiv preprint arXiv:2604.15218*, 2026. doi:10.48550/arXiv.2604.15218.

18 Rohan Goyal, Venkatesan Guruswami, Yihang Sun, and Mary Wootters. Locality of curve-decoding and improved proximity gaps, 2026. arXiv:2607.08516.

19 Kobi Gurkan, Andrija Novakovic, and Ron D Rothblum. Bolt: Faster snarks from sketched codes. *Cryptology ePrint Archive*, 2026.

20 Venkatesan Guruswami and Swastik Kopparty. Explicit subspace designs. *Combinatorica*, 36(2):161–185, 2016. doi:10.1007/s00493-014-3169-1.

21 Venkatesan Guruswami, Ray Li, Jonathan Mosheiff, Nicolas Resch, Shashwat Silas, and Mary Wootters. Bounds for list-decoding and list-recovery of random linear codes. *IEEE Transactions on Information Theory*, 68(2):923–939, 2021. doi:10.1109/TIT.2021.3127126.

22 Venkatesan Guruswami, Jonathan Mosheiff, Nicolas Resch, Shashwat Silas, and Mary Wootters. Threshold rates for properties of random codes. *IEEE Transactions on Information Theory*, 68(2):905–922, 2021. doi:10.1109/TIT.2021.3123497.

23 Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on information theory*, 54(1):135–150, 2008. doi:10.1109/TIT.2007.911222.

24 Ulrich Haböck. Basefold in the list decoding regime. Cryptology ePrint Archive, Paper 2024/1571, 2024. URL: https://eprint.iacr.org/2024/1571.

25 Swastik Kopparty, Noga Ron-Zewi, Shubhangi Saraf, and Mary Wootters. Improved decoding of folded reed-solomon and multiplicity codes. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 212–223. IEEE, 2018. doi:10.1109/FOCS.2018.00029.

26 Dmitry Krachun, Stepan Kazanin, and Ulrich Haböck. Failure of proximity gaps close to capacity. Cryptology ePrint Archive, Paper 2026/782, 2026. URL: https://eprint.iacr.org/2026/782.

27 Matan Levi, Jonathan Mosheiff, and Nikhil Shagrithaya. Random reed-solomon codes and random linear codes are locally equivalent. In *2025 IEEE 66th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 2097–2131. IEEE, 2025. doi:10.1109/FOCS63196.2025.00112.

28 Jonathan Mosheiff, Nicolas Resch, Noga Ron-Zewi, Shashwat Silas, and Mary Wootters. Low-density parity-check codes achieve list-decoding capacity. *SIAM Journal on Computing*, 53(6):FOCS20–38–FOCS20–73, November 2021. doi:10.1137/20m1365934.

29 Guy N Rothblum, Salil Vadhan, and Avi Wigderson. Interactive proofs of proximity: delegating computation in sublinear time. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 793–802, 2013. doi:10.1145/2488608.2488709.

30 Shashank Srivastava. Improved list size for folded reed-solomon codes. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2040–2050. SIAM, 2025. doi:10.1137/1.9781611978322.64.

31 Itzhak Tamo. Tighter list-size bounds for list-decoding and recovery of folded reed-solomon and multiplicity codes. *IEEE Transactions on Information Theory*, 70(12):8659–8668, 2024. doi:10.1109/TIT.2024.3402171.

Learning and Testing Convex Functions

Renato Ferreira Pinto Jr.
Columbia University, New York, NY, USA

Cassandra Marcussen
Harvard University, Cambridge, MA, USA

Elchanan Mossel
MIT, Cambridge, MA, USA

Shivam Nadimpalli
MIT, Cambridge, MA, USA

Abstract

We consider the problems of *learning* and *testing* real-valued convex functions over Gaussian space. Despite the extensive study of function convexity across mathematics, statistics, and computer science, its learnability and testability have largely been examined only in discrete or restricted settings – typically with respect to the Hamming distance, which is ill-suited for real-valued functions.

In contrast, we study these problems in high dimensions under the standard Gaussian measure, assuming sample access to the function and a mild smoothness condition, namely Lipschitzness. A smoothness assumption is natural and, in fact, necessary even in one dimension: without it, convexity cannot be inferred from finitely many samples. As our main results, we give:

- Learning Convex Functions: An agnostic proper learning algorithm for Lipschitz convex functions that achieves error ε using $n^{O(1/\varepsilon^2)}$ samples, together with a complementary lower bound of $n^{\mathrm{poly}(1/\varepsilon)}$ samples in the *correlational statistical query (CSQ)* model.
- Testing Convex Functions: A tolerant (two-sided) tester for convexity of Lipschitz functions with the same sample complexity (as a corollary of our learning result), and a one-sided tester (which never rejects convex functions) using $O(\sqrt{n}/\varepsilon)^n$ samples.

2012 ACM Subject Classification Theory of computation → Sample complexity and generalization bounds; Theory of computation → Models of learning; Computing methodologies → Supervised learning by regression; Theory of computation → Convex optimization

Keywords and phrases Property testing, learning theory, Gaussian distribution, convex functions

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.62

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2511.11498`

Funding *Renato Ferreira Pinto Jr.*: Supported by an NSERC Postdoctoral Fellowship and by NSF grant CCF-2106429.
Cassandra Marcussen: This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-23-F-0014 (NDSEG Fellowship). Supported in part by NSF Award 2152413 and a Simons Investigator Award to Madhu Sudan.
Elchanan Mossel: Supported by ARO MURI N000142412742, by NSF grant DMS-2031883, by Vannevar Bush Faculty Fellowship ONR-N00014-20-1-2826 and by a Simons Investigator Award.

Acknowledgements The authors would like to thank Ramon van Handel for several enjoyable discussions concerning Question 5. E.M. would like to thank Dan Mikulincer for interesting discussions.



© Renato Ferreira Pinto Jr., Cassandra Marcussen, Elchanan Mossel, and Shivam Nadimpalli;
licensed under Creative Commons License CC-BY 4.0



1 Introduction

Few mathematical ideas are as pervasive or as powerful as convexity: recall that a function $f : \mathbb{R}^n \to \mathbb{R}$ is *convex* if for every $x, y \in \mathbb{R}^n$ and $\lambda \in [0, 1]$, we have $f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$. The mathematics of convex functions has been intensively studied for over a century, and continues to be central to a number of theoretical and applied disciplines ranging from mathematical analysis to machine learning [68, 15]. Within mathematical programming, for example, convexity of the objective function often governs computational tractability; to quote Rockafellar [69], "the great watershed in optimization isn't between linearity and nonlinearity, but convexity and non-convexity."

This centrality of convexity motivates the following basic questions, which form the focus of this work:

How many samples are needed to *learn* a convex function?

How many samples are needed to *test* if a function $f : \mathbb{R}^n \to \mathbb{R}$ is essentially convex?

We study each of these questions through the lens of theoretical computer science, viewing the former as a problem in *learning theory* [48] and the latter in *property testing* [36, 7]. For these problems to be well-defined, we must first specify a distance metric between functions on $\mathbb{R}^n$. Throughout this work, we employ the $L^2(\gamma)$ distance where $\gamma = \gamma_n$ denotes the measure corresponding to the n-dimensional standard Gaussian distribution $N(0, I_n)$.

Unsurprisingly, given their fundamental nature, both problems have been studied previously:

- There is work in statistics [72, 40, 41, 61, 32] that deals with optimal rates for convex regression where the perspective is that the dimension is fixed, and the number of samples is very large (could be more than exponential) and the question is to study the error rate as the number of samples goes to infinity. This is different than the standard perspective in learning theory where we are interested to more exactly quantify the number of samples needed as a function of the dimension.
- Turning to the problem of *testing* convexity, prior work has largely concerned discrete or highly constrained settings. Much of this literature [65, 6, 63, 5, 4, 54] focuses on testing functions over finite domains or ranges under the Hamming distance, reflecting the field's early connections to complexity theory [13, 71]. A few works such as [6] extend this line of work to real-valued functions under L^p metrics, but still assume bounded range and full query access.

We refer the reader to Section 1.2 for a detailed discussion of related work on both testing and learning convex functions.

In this work, we study both *agnostic proper learning* and *(tolerant) testing* of convexity in high dimensions, in the statistically natural setting where one only has access to *samples* labeled by an unknown function f. The query-access model used extensively in property testing, while natural and powerful for applications to complexity theory [13, 71], is ill-suited to statistical problems. By contrast, sample-based access is the standard framework in settings such as empirical risk minimization, convex regression, and stochastic optimization, where functions are observed only through (possibly noisy) random samples rather than arbitrary queries [67, 51, 44, 14, 73, 39].

Before proceeding, we note that some smoothness assumption is essential for both problems: without any regularity, convexity cannot be meaningfully inferred from finitely many samples, even in one dimension. Accordingly, we restrict attention to *Lipschitz* functions – a natural and mild assumption also adopted in prior work on convex regression [61].

1.1 Our Results

We begin by introducing notation that will be used throughout the paper. Given a function $f \in L^2(\gamma)$, we will write $\mathrm{d}^L_{\mathrm{conv}}(f)$ for the $L^2(\gamma)$ distance of f to its nearest convex L-Lipschitz function. More formally,

$$\mathrm{d}^L_{\mathrm{conv}}(f) := \inf_{\substack{g:\mathbb{R}^n\to\mathbb{R} \\ g \text{ convex, } L\text{-Lipschitz}}} \|f-g\|_{L^2(\gamma)}\,,$$

where

$$\|f-g\|_{L^2(\gamma)} := \mathop{\mathbf{E}}_{\boldsymbol{x}\sim N(0,I_n)}\left[(f(\boldsymbol{x})-g(\boldsymbol{x}))^2\right]^{1/2}.$$

The $L^2(\gamma)$ metric is a standard choice in learning theory, property testing, and derandomization (see, for example, [52, 60, 78, 43, 53, 20, 22, 28, 23, 49, 26, 27], among many others).

Although we focus on learning and testing convex functions over Gaussian space, all of our techniques and results extend naturally to functions over $[0,1]^n$ under the uniform measure. We adopt the Gaussian setting as it is a more canonical model for studying real valued learning and testing problems.

1.1.1 Agnostic Proper Learning and Sample-Based Testing of Convex Functions

We construct an agnostic proper learning algorithm for Lipschitz convex functions with the following guarantees:

▶ **Theorem 1** (Agnostic proper learning of Lipschitz convex functions). *Let $\varepsilon, L > 0$ and suppose $f:\mathbb{R}^n\to\mathbb{R}$ is a L-Lipschitz function. There exists an algorithm which, given i.i.d. access to labeled samples $(\boldsymbol{x}, f(\boldsymbol{x}))$ where $\boldsymbol{x}\sim N(0,I_n)$, draws $n^{O(L^2/\varepsilon^2)}$ samples, runs in time $\exp(\widetilde{O}(nL^2/\varepsilon^2))$, and outputs a function $g\in L^2(\gamma)$ such that*

$$\|f-g\|_{L^2(\gamma)} \le \mathrm{d}^L_{\mathrm{conv}}(f)+\varepsilon\,.$$

Additionally, the function g is convex and L-Lipschitz.

Note that our learning algorithm is agnostic in the sense that it produces a nearly optimal (convex) hypothesis even if the input function f is arbitrarily far from convex, as long as it satisfies the Lipschitz condition.

Using the proof of the Theorem above and the "testing-via-learning" paradigm [37], we obtain a *tolerant* testing algorithm for Lipschitz convex functions:

▶ **Theorem 2** (Tolerant two-sided testing of Lipschitz convex functions). *Let $\varepsilon, \varepsilon_0, L \ge 0$ and suppose $f:\mathbb{R}^n\to\mathbb{R}$ is L-Lipschitz. There is an algorithm which, given i.i.d. labeled samples $(\boldsymbol{x}, f(\boldsymbol{x}))$ where $\boldsymbol{x}\sim N(0,I_n)$, draws $n^{O(L^2/\varepsilon^2)}$ samples, runs in time $\exp(\widetilde{O}(nL^2/\varepsilon^2))$, and has the following performance guarantee:*

- *If $\mathrm{d}^L_{\mathrm{conv}}(f)\le\varepsilon_0$, then it outputs "accept" with probability $9/10$;*
- *If $\mathrm{d}^L_{\mathrm{conv}}(f)\ge\varepsilon+\varepsilon_0$, then it outputs "reject" with probability $9/10$.*

In particular, setting $\varepsilon_0 = 0$ shows that for every fixed $\varepsilon > 0$, there exists an (essentially) $\exp(n)$-time algorithm that distinguishes Lipschitz convex functions from those that are ε-far (in $L^2(\gamma)$) from all Lipschitz convex functions. The "testing-via-learning" approach underlies

the best known sample-based testing algorithms for several other function classes including convex subsets of $\mathbb{R}^n$ [52, 20], monotone Boolean functions [17, 56, 58], and k-monotone Boolean functions [8].

Our proof of Theorem 1 proceeds in two steps:

1. **Low-Degree Approximation.** We begin with a standard observation (see, for example, Theorem 4.1 of [2] or alternatively [46, 23]) that every L-Lipschitz function over Gaussian space is well-approximated in L_2 by a $O(L^2)$-degree polynomial. This allows us to learn such an approximation $\widetilde{f}$ from random samples via polynomial regression.
2. **Convex Regression.** We then *convexify* the learned polynomial $\widetilde{f}$ by projecting it onto the space of convex functions via convex regression [61]. This step is essential to obtain a *proper* learner – one whose output lies within the target class of convex L-Lipschitz functions. For our tolerant testing result, this is essential: the "testing via learning" reduction requires the learner to be proper.

Although the first step is conceptually straightforward, it has an independent implication:

> Any $O(1)$-Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}$ can be learned – and hence tested – from $n^{O(1/\varepsilon^2)}$ random samples in an information-theoretic sense.

Note, however, that this guarantee is purely information-theoretic and does not yield a computationally efficient learning algorithm for the function class. In general, however, fitting a function from the target class to the observed samples may be computationally intractable. The main conceptual contribution of Theorem 1 is that we may combine standard low-degree learning with tools specific to convexity to obtain a *proper* learning algorithm.

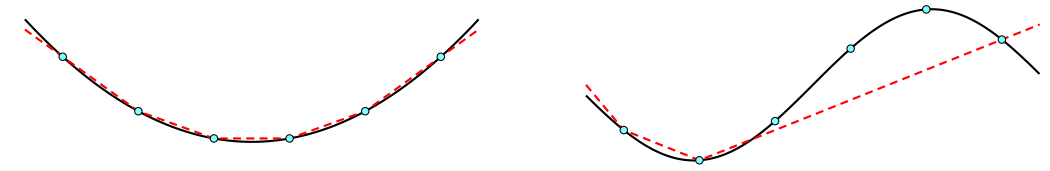
Accordingly, the second step above introduces the main technical challenge: ensuring that the projection onto convex functions can be performed in finite (albeit exponential) time while preserving the $L^2(\gamma)$ error bound. Using queries to the learned low-degree approximation $\widetilde{f}$ on a sufficiently fine grid, we learn via convex regression a function $\widetilde{g}$ (namely a supremum of affine functions) which approximates $\widetilde{f}$ well on a discrete grid. Following [61], quadratic constraints in the convex regression ensure that $\widetilde{g}$ is L-Lipschitz. However, convex regression says nothing about the quality of the approximation over (continuous) Gaussian space, and the low-degree $\widetilde{f}$ need not inherit the Lipschitz behavior of f. To bridge this gap, we draw on Hermite analysis to show that the low-degree approximation $\widetilde{f}$ is L'-Lipschitz for some $L' \gg L$ over a large box in $\mathbb{R}^n$, where both L' and the size of the box can be controlled in terms of n, L and ε. This lets us recover an $L^2(\gamma)$ approximation guarantee for the convex $\widetilde{g}$ against the original target function f.

1.1.2 Lower Bounds for Learning Convex Functions

We complement this upper bound with corresponding *lower bounds* for learning convex functions. In the information-theoretic setting, we show that learning an arbitrary linear function to constant accuracy requires $\Omega(n)$ queries, not just samples; see the full version. We further strengthen this by proving a quantitatively sharper bound for algorithms in the *correlational statistical query* (CSQ) model:

▶ **Theorem 3** (CSQ lower bound; informal, see full version)**.** *Suppose algorithm A, given access to L-Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}$ via correlational statistical queries, outputs a function $g \in L^2(\gamma)$ satisfying $\|f - g\|_{L^2(\gamma)} \le \mathrm{d}^L_{\mathrm{conv}}(f) + \varepsilon$. Then A requires $n^{\mathrm{poly}(L/\varepsilon)}$ correlational statistical queries with tolerance $n^{-\mathrm{poly}(L/\varepsilon)}$.*

CSQ algorithms only have access to the input f via approximate inner product queries of the form $\mathbf{E}_{\boldsymbol{z}\sim N(0,I_n)}[f(\boldsymbol{z})h(\boldsymbol{z})]$, for query h chosen by the algorithm; we formally define the CSQ model in the full version. This model captures, for example, gradient descent



(a) (b)

Figure 1 One-dimensional illustration of the empirical convex envelope (red, dashed) of (a) a convex function and (b) a far-from-convex function, with respect to the sampled points shown in cyan. (See Definition 19 for a precise definition).

with respect to squared loss, and the algorithm attaining our $n^{O(L^2/\varepsilon^2)}$ upper bound from Theorem 1 via low-degree approximation is a CSQ algorithm. Thus, Theorem 3 poses a barrier to improving upon the low-degree learning approach: any agnostic learning algorithm that fits the CSQ model, even if it uses properties of convex functions to do something more sophisticated than low-degree approximation, must use $n^{\mathrm{poly}(L/\varepsilon)}$ statistical queries.

Our proof relies on the construction of a large family $\mathcal{F}$ of functions with low pairwise-correlation, which is the standard approach for showing *weak learning* lower bounds in the CSQ model [12, 75, 33]. Additionally, we require each function in $\mathcal{F}$ to be non-trivially correlated with a convex Lipschitz function – namely a "projected ReLU", which is a function of the form $x \mapsto \mathrm{ReLU}(\langle x, u\rangle)$ for some $u \in \mathbb{S}^{n-1}$. This implies that an agnostic learning algorithm for convex functions gives a weak learner for $\mathcal{F}$, and yields our CSQ lower bound.

Our construction and proof strategy are inspired by and leverage the results of [30], who showed an $n^{\mathrm{poly}(1/\varepsilon)}$ lower bound for agnostically learning ReLUs under the Gaussian distribution using statistical queries. While one might hope to immediately conclude the same lower bound for agnostically learning convex functions in our model (since ReLUs are convex), the key difference is that in our setting the input must be Lipschitz, whereas [30] build their hard family of inputs from Boolean functions. We adapt their construction to our purposes via careful application of the Ornstein-Uhlenbeck noise operator P_t, which satisfies a few crucial properties for our application:

- It takes bounded functions into Lipschitz ones, so that from a hard input $f : \mathbb{R}^n \to \{\pm 1\}$, we may obtain a hard Lipschitz input $P_t f$.
- It satisfies the self-adjointness property $\langle P_t f, g\rangle = \langle f, P_t g\rangle$, which allows us to map the approximation quality of a learned hypothesis g for the Lipschitz input $P_t f$ into the approximation quality of hypothesis $P_t g$ for the hard input f.
- Lipschitz functions are stable under noise in the sense that $\|P_t f - f\|_2^2 = O(t)$, which informally implies that most of the "signal" in the reduction survives our application of noise.

1.1.3 One-Sided Testing of Lipschitz Convex Functions

Our final result is a *one-sided* testing algorithm for Lipschitz convex functions – that is, an algorithm which never rejects a Lipschitz convex function. In general, there can be dramatic separations between the complexity of one-sided and two-sided testing (cf. [37, §1.5]), and in particular, testing by learning is not a suitable approach to one-sided testing. Therefore, this goal requires us to exploit the properties of convex and Lipschitz functions in a different way. Our result is:

▶ **Theorem 4** (One-sided testing of Lipschitz convex functions). *Let $f : \mathbb{R}^n \to \mathbb{R}$ be L-Lipschitz, and let $\varepsilon \geq 0$. There is an algorithm, One-Sided-Tester (Algorithm 1), which, given access to i.i.d. labeled samples $(\boldsymbol{x}, f(\boldsymbol{x}))$ where $\boldsymbol{x} \sim N(0, I_n)$, makes $O(\sqrt{n}L/\varepsilon)^n$ draws, runs in time $(nL/\varepsilon)^{O(n)}$, and has the following performance guarantee:*

- *If f is convex, then One-Sided-Tester outputs "accept" with probability 1;*
- *If $\mathrm{d}^L_{\mathrm{conv}}(f) \geq \varepsilon$, then One-Sided-Tester outputs "reject" with probability $9/10$.*

Our tester operates by constructing an *empirical convex envelope* (cf. Definition 19) of the sampled function values, which can be viewed as the function analogue of the convex hull of a collection of points. Given random samples $(\boldsymbol{x}_i, f(\boldsymbol{x}_i))$, the tester checks whether f coincides with its empirical convex envelope at these points. If f is convex, this always holds; conversely, we show that if f is L-Lipschitz and ε-far from convex, then with sufficiently many samples the test will detect a violation with high probability. See Figure 1 for an illustration of the empirical convex envelope.

In particular, our analysis implies an associated *agnostic learning guarantee in $L^\infty(\gamma)$*: we show that when the input function f is an L-Lipschitz convex function, the empirical convex envelope learned by One-Sided-Tester uniformly approximates the convex envelope of f over all but an exponentially-small fraction of Gaussian space. Thus, the algorithm may also be viewed as a form of *uniform convex regression*.

1.2 Related Work

We briefly situate our results within the broader work on learning and testing convexity.

Learning Convex Functions

As discussed earlier, the statistical perspective for *convex regression* is interested in the error rate as the number of samples goes to infinity for fixed dimensions. This falls under the broader umbrella of *shape-constrained regression*. We refer the interested reader to [72, 40, 41, 61, 32] and the references therein for further background. By contrast, our results address the more general problem of *agnostic proper learning* from i.i.d. samples, with the goal of establishing PAC-style sample-complexity bounds rather than asymptotic error rates.

We note that low-degree approximation of Lipschitz functions over Gaussian space has been used a few times in the past; see, for example, [2, 46, 23]. Such approximations also are the source of $n^{\mathrm{poly}(1/\varepsilon)}$ sample-complexity bounds appearing in several agnostic learning problems beyond convexity, e.g., in the context of learning halfspaces [31, 57]. However, with the exception of [2], most of these works focus on Boolean-valued function classes rather than real-valued ones.

Approximation Theory

As indicated earlier in Section 1.1, our upper bounds for convexity testing follow from semi-agnostic learning algorithms for convex functions under the Gaussian L_2 distance. Related problems have been studied in approximation theory, both in the low-dimensional setting [74, 70] as well as in high-dimensions [45], with exponential-in-n sample lower bounds known for deterministic approximation under L^p distance [45].

More broadly, classical results in convex approximation typically focus on deterministic settings, approximating a convex function by piecewise-linear or polynomial surrogates and bounding the error in terms of smoothness or curvature (see, e.g., [29, 45]). Our setting differs in that we consider randomized, sample-based approximation under a fixed measure, leading to dimension-dependent but probabilistic guarantees.

Testing Convexity over Discrete Domains

The computational problem of recognizing convexity of quartic polynomials is known to be NP-hard [1]. (Deciding convexity of polynomials of odd-degree or quadratics is trivial, making degree 4 the first interesting case.) See [1] and the references therein for a history of the computational problem of recognizing *exact* convexity of functions. The relaxed problem of recognizing *approximate* convexity of functions has primarily been considered in the discrete setting. The study of testing convexity of functions $f : [N] \to \mathbb{R}$ under the Hamming distance was initiated by Parnas, Ron, and Rubinfeld [64], where convexity is defined as non-negativity of a discrete second-derivative. Their was later extended by Ben-Eliezer [5], who obtained improved quantitative bounds; by Pallavoor, Raskhodnikova, and Verma [63], who gave an improved bound parametrized by the size of the range of the function being tested; by Blais, Raskhodnikova, and Yaroslavtsev [11], who gave lower bounds on testing convexity of real-valued functions over the hypergrid $[N]^d$; and by Belovs, Blais, and Bommireddi [4], who gave upper and lower bounds on the number of queries required to test convexity of real-valued functions over various discrete domains including the discrete line, the "stripe" $[3] \times N$, and the hypergrid $[N]^d$. These results, however, do not have any bearing on this work due to differences in domain and choice of distance metric.

L^p Testing of Convexity

The work most closely related to ours is that of Berman, Raskhodnikova, and Yaroslavtsev [6], who study L^p testing of convexity of an unknown $f : [0,1]^n \to [0,1]$ where $[0,1]^n$ is endowed with the uniform measure. They provide tight bounds in the one-dimensional setting and exponential-in-n bounds for the high-dimensional case (see Corollary B.3 of [6]). Our results, in contrast, apply specifically to Lipschitz functions $f : \mathbb{R}^n \to \mathbb{R}$, giving polynomial-in-$n$ complexity algorithms for every fixed choice of the distance parameter ε. (We remark that our exponential-in-n algorithm from Theorem 4 has the additional property of being a tester with one-sided error, which does not follow from a black-box application of testing-by-learning.)

Although one might hope to relate the two settings, this connection does not appear to be straightforward. The two frameworks are best viewed as complementary: ours imposes regularity assumptions that enable polynomial dependence on n, whereas [6] addresses unrestricted (i.e., possibly highly irregular) albeit bounded functions at exponential cost. Indeed, assuming Lipschitzness or bounded gradient is the standard regularity condition in convex analysis, optimization, and high-dimensional learning, and thus provides a more natural setting for analytic property testing.

Testing Set Convexity

A different line of research considers testing convexity of high-dimensional sets (i.e., Boolean-valued indicator functions of subsets of $\mathbb{R}^n$). Rademacher and Vempala [66] established exponential-in-n upper bounds for testing set convexity under the Lebesgue measure; they also gave a strong lower bound against a simple "line" tester for convexity. The [66] lower bound was subsequently strengthened by Blais and Bommireddi [10]. For testing set convexity under the Gaussian measure, the best-known upper bound follows from the agnostic learning algorithm for convex sets due to Klivans, O'Donnell, and Servedio [52].

Strong lower bounds for sample-based testing under the Gaussian measure were established by Chen, Freilich, Servedio, and Sun [20], and the first lower bounds in the query model were obtained recently by Chen, De, Nadimpalli, Servedio, and Waingarten [21]. While our work shares conceptual and technical similarities with [52, 20, 21], there does not appear to be a

reduction or formal connection between the two settings. In particular, natural attempts to reduce testing convexity of functions to testing convexity of sets (such as mapping a function to its epigraph) fail to preserve the relevant distance measures, so neither problem appears to subsume the other.

Finally, beyond the Gaussian setting, the work of Black, Blais, and Harms [9] also considers consider testing and learning convex subsets of the ternary hypercube $\{0, \pm 1\}^n$.

1.3 Discussion

As evidenced above, convexity is a powerful and pervasive principle whose associated computational problems are notoriously challenging and enjoy long lines of study from a plethora of perspectives. We view the present work as a first step in posing and investigating the problems of learning and testing convex functions in the continuous setting from the perspective of learning theory and property testing. Our results establish that these problems are computationally feasible, albeit not at present with efficient algorithms, and suggest barriers to be tackled for overcoming this limitation: any agnostic learning algorithm with truly $\mathrm{poly}(n)$ sample complexity, beating the low-degree learning bound from Theorem 1, must not be a CSQ algorithm (Theorem 3); and any algorithm (for learning or testing) with subexponential running time must bypass the grid-based approaches of Theorems 1 and 4.

We conclude with a number of open directions suggested by our work.

Analogy with Boolean Monotonicity

Convexity is formally connected to the monotonicity of the gradient: in one dimension, a function is convex if and only if its derivative is monotone, and more generally, convexity corresponds to the monotonicity of the gradient map. This connection suggests a natural analogy between convexity and monotonicity, and one might hope that insights from the rich body of work on learning and testing monotone Boolean functions can inform the study of convex function learning and testing.

For *learning*, our results may at first appear considerably stronger than existing bounds for learning monotone Boolean functions. (Recall that the sample complexity of agnostically learning monotone Boolean functions over $\{0,1\}^n$ is $2^{\widetilde{O}(\sqrt{n})}$ [17].) However, this stems from the $O(1)$-Lipschitz assumption in our setting, which has no meaningful analog in the Boolean function setting. While all Boolean functions are technically $O(1)$-Lipschitz, the underlying metric geometry differs between the Boolean setting and Gaussian space: in the Boolean space (see, for example, [55]), the typical L_1 (Hamming) distance between points in dimension n is of order n while in the Gaussian space the typical distance is of order $\sqrt{n}$. So, at least in some sense, our polynomial agnostic learning algorithm does correspond to the results for learning monotone functions with $2^{\Theta(\sqrt{n})}$ samples.

As for *testing*, in the Boolean setting the analogous quantity to $\mathrm{d}_{\mathrm{conv}}$ is the *distance to monotonicity* $\mathrm{d}_{\mathrm{mon}}$, which has been extensively investigated in a rich line of work on monotonicity testing [38, 18, 50]. Classical Boolean monotonicity testers rely on upper bounds on $\mathrm{d}_{\mathrm{mon}}$ in terms of the *directed edge boundary*, and these structural relationships can be viewed as strengthenings of the classical edge-isoperimetric inequalities on the hypercube [59, 76].

This analogy raises a natural question: can $\mathrm{d}_{\mathrm{conv}}$ likewise give rise to structural inequalities that quantify the "non-convexity" of real-valued functions? Motivated by the role of the directed edge boundary in monotonicity testing, the following question suggests itself:

▶ **Question 5.** *Given an L-Lipschitz function $f : \mathbb{R}^n \to \mathbb{R}$, is*

$$\mathrm{d}_{\mathrm{conv}}(f) \leq \mathrm{poly}(n) \cdot \mathop{\mathbf{E}}_{\boldsymbol{x} \sim N(0,I_n)} \left[\left\| \nabla^2_- f(\boldsymbol{x}) \right\|_{\mathrm{op}} \right]$$

where $\nabla^2_- f(\boldsymbol{x})$ is the negative-definite part of $\nabla^2 f(\boldsymbol{x})$?

If true, such an inequality would likely imply a convexity tester based on random samples of (or queries simulating) the Hessian of f, mirroring how monotonicity testers – both in discrete and continuous settings – leverage local gradient information [50, 34]. Beyond its algorithmic implications, this inequality would also be of intrinsic geometric interest, as it would relate a *global* measure of non-convexity (namely, $\mathrm{d}_{\mathrm{conv}}(f)$) to *local* curvature data through the Hessian.

As a step toward understanding $\mathrm{d}_{\mathrm{conv}}$, we establish in the full version a quantitative relationship between $\mathrm{d}_{\mathrm{conv}}(f)$ and the negative degree-2 Hermite coefficients of f, which yields a lower bound on $\mathrm{d}_{\mathrm{conv}}(f)$ in terms of their magnitude.

We note that an independent line of work has identified an unexpected but powerful analogy between monotone Boolean functions and *convex subsets of Gaussian space* [24, 28, 25, 26, 27]. It is natural to ask which aspects of this correspondence extend further to the setting of convex functions over Gaussian space. As a first step in this direction, we note that [24] establishes a quantitative sharpening of Hu's correlation inequality for symmetric convex sets [47], in the spirit of Talagrand's quantitative correlation inequality [77].

Testing Lower Bounds

Since we are concerned with functions $f : \mathbb{R}^n \to \mathbb{R}$ whose outputs are real-valued, standard information-theoretic approaches to proving lower bounds become infeasible. A basic but significant obstacle – shared with many real-valued testing problems – is that each sample reveals a real number $f(x)$, which, at least naively, may contain an arbitrarily large amount of information. Presently, there is no general framework for establishing lower bounds for testing convexity of real-valued functions, even in the purely sample-based setting.

Therefore, we pose the following two questions on the true sample complexity of testing convexity of Lipschitz functions in the two-sided and one-sided error regimes. For two-sided testing, it is conceivable – again in analogy with monotonicity testing – that some polynomial dependence on n is unavoidable. We ask:

▶ **Question 6.** *Does testing convexity of 1-Lipschitz functions under the Gaussian distribution (with two-sided error) require $n^{\Omega(1)}$ samples? What if arbitrary queries are allowed?*

Even for one-sided testing algorithms, we currently lack any nontrivial lower bounds on the sample complexity of testing convexity. It is instructive to recall how an exponential lower bound is obtained for the analogous problem in the setting of *set* convexity [20]. In particular, Lemma 29 of [20] establishes that for sufficiently small constant c, if 2^{cn} points are drawn independently from $N(0, I_n)$, then with high probability no point lies in the convex hull of the others. Consequently, such a draw of samples is consistent with the indicator function of a convex set, and a one-sided sample-based tester cannot detect any violation of convexity.

The difficulty in our setting is that a sample with no witnessed violation of convexity need not be *consistent* with any L-Lipschitz function that is convex. In particular, a naive interpolation through the sampled points may fail to be $O(L)$-Lipschitz. Thus, it is not immediate that a one-sided tester must accept unless the sample explicitly contains a convexity violation. This leads to the following question:

▶ **Question 7.** *Is there a tester for convexity of* 1*-Lipschitz functions under the Gaussian distribution with one-sided error and sample complexity that is sub-exponential in* n*?*

Sample vs. Runtime Gaps

We note that while Theorem 1 achieves a sample complexity of $n^{O(1/\varepsilon^2)}$, its runtime remains exponential in n. An analogous gap long persisted for properly learning monotone Boolean functions from samples: the best-known sample complexity was $2^{\widetilde{O}(\sqrt{n})}$, whereas the best runtime was $2^{O(n)}$. Recent work by [56, 58] closed this gap, giving a $2^{\widetilde{O}(\sqrt{n})}$-time agnostic proper learner. It is natural to ask whether a similarly efficient proper learning algorithm exists for convex functions, potentially closing the gap between information-theoretic and computational efficiency in our setting as well.

Lower Bounds in the Full SQ Model

Our $n^{\mathrm{poly}(L/\varepsilon)}$ lower bound from Theorem 3 applies to correlational statistical query (CSQ) algorithms, a setting that already captures the algorithm attaining our $n^{O(L^2/\varepsilon^2)}$ upper bound. Recall that CSQ algorithms only have access to the input f via approximate inner product queries of the form $\mathbf{E}_{\boldsymbol{z}\sim N(0,I_n)}[f(\boldsymbol{z})h(\boldsymbol{z})]$. In contrast, the full SQ model allows approximate oracle access to $\mathbf{E}_{\boldsymbol{z}\sim N(0,I_n)}[q(\boldsymbol{z}, f(\boldsymbol{z}))]$ for general query functions $q(x, y)$.

In the (distribution-specific) setting of *Boolean* functions, the CSQ model is equivalent to the full SQ model [16], because SQ queries to Boolean functions may be simulated using correlational queries. However, the same is not true in the real-valued setting, where there exist separations between the CSQ and SQ models, for example for the problem of ReLU regression [42]. Indeed, proving lower bounds against SQ algorithms in the real-valued setting is a notoriously challenging problem; see [35, 19] and the discussion therein. We leave as an open question whether our $n^{\mathrm{poly}(L/\varepsilon)}$ CSQ lower bound can be extended to the full SQ model, or whether our problem witnesses a separation between these models.

Organization

In the rest of this paper, we outline the proofs of our main upper bound results – Theorems 1, 2, and 4. We defer proofs of technical lemmas and our lower bounds to the full version.

2 Preliminaries

We use boldfaced letters – such as $\boldsymbol{x}, \boldsymbol{f}, \boldsymbol{A}$ – to denote random variables (which may be real-valued, vector-valued, function-valued, or set-valued; the intended type will be clear from the context). We write $\boldsymbol{x} \sim \mathcal{D}$ to indicate that the random variable $\boldsymbol{x}$ is distributed according to the probability distribution $\mathcal{D}$. Throughout, $\mathbb{S}^{n-1} := \{x \in \mathbb{R}^n : \|x\|_2 = 1\}$ will denote the unit sphere in $\mathbb{R}^n$. We will write $B_p(r) \subseteq \mathbb{R}^n$ for the ℓ_p-ball of radius r, i.e.,

$$B_p(r) := \{x \in \mathbb{R}^n : \|x\|_p \leq r\}.$$

Hermite Analysis over Gaussian Space

Our notation and terminology follow Chapter 11 of [62]. We say that an n-dimensional *multi-index* is a tuple $\alpha \in \mathbb{N}^n$, and we define $|\alpha| := \sum_{i=1}^n \alpha_i$.

For $n \in \mathbb{N}_{>0}$, we write $L^2(\gamma_n)$ to denote the space of functions $f : \mathbb{R}^n \to \mathbb{R}$ that have finite second moment under the Gaussian distribution, i.e., $f \in L^2(\gamma_n)$ if

$$\|f\|^2_{L^2(\gamma_n)} := \mathop{\mathbf{E}}_{\boldsymbol{z} \sim N(0,I_n)} \left[f(\boldsymbol{z})^2\right] < \infty \,.$$

When the dimension n is clear from context, we will write $\gamma = \gamma_n$ instead. We view $L^2(\gamma)$ as an inner product space with

$$\langle f, g \rangle_{L^2(\gamma)} := \mathop{\mathbf{E}}_{\boldsymbol{z} \sim N(0,I_n)} [f(\boldsymbol{z}) g(\boldsymbol{z})] \,.$$

We will sometimes write $\langle f, g \rangle = \langle f, g \rangle_{L^2(\gamma)}$ for notational convenience. We recall the Hermite basis for $L^2(\gamma_1)$:

▶ **Definition 8** (Hermite basis). *The* Hermite polynomials $(h_j)_{j \in \mathbb{N}}$ *are the univariate polynomials defined as*

$$h_j(x) = \frac{(-1)^j}{\sqrt{j!}} \exp\left(\frac{x^2}{2}\right) \cdot \frac{d^j}{dx^j} \exp\left(-\frac{x^2}{2}\right).$$

In particular, we have

$$h_2(x) := \frac{x^2 - 1}{\sqrt{2}} \,. \tag{1}$$

The following fact is standard:

▶ **Fact 9** (Proposition 11.33 of [62]). *The Hermite polynomials $(h_j)_{j \in \mathbb{N}}$ form a complete, orthonormal basis for $L^2(\gamma_1)$. For $n > 1$ the collection of n-variate polynomials given by $(h_\alpha)_{\alpha \in \mathbb{N}^n}$ where*

$$h_\alpha(x) := \prod_{i=1}^{n} h_{\alpha_i}(x)$$

forms a complete, orthonormal basis for $L^2(\gamma_n)$.

Given a function $f \in L^2(\gamma)$ and $\alpha \in \mathbb{N}^n$, we define its *Hermite coefficient on* α as $\widehat{f}(\alpha) = \langle f, h_\alpha \rangle$. It follows that $f : \mathbb{R}^n \to \mathbb{R}$ can be uniquely expressed as

$$f = \sum_{\alpha \in \mathbb{N}^n} \widehat{f}(\alpha) h_\alpha$$

with the equality holding in $L^2(\gamma)$; we will refer to this expansion as the *Hermite expansion* of f. One can check that Parseval's and Plancharel's identities hold in this setting:

$$\langle f, f \rangle = \sum_{\alpha \in \mathbb{N}^n} \widehat{f}(\alpha)^2 \qquad \text{and} \qquad \langle f, g \rangle = \sum_{\alpha \in \mathbb{N}^n} \widehat{f}(\alpha) \widehat{g}(\alpha).$$

We recall the Gaussian Poincaré inequality (see, e.g., [3]):

▶ **Proposition 10.** *Suppose $f \in L^2(\gamma)$ is differentiable. Then*

$$\mathop{\mathbf{Var}}_{\boldsymbol{x} \sim N(0,I_n)} [f(\boldsymbol{x})] \leq \mathop{\mathbf{E}}_{\boldsymbol{x} \sim N(0,I_n)} \left[\|f(\boldsymbol{x})\|_2^2\right] .$$

3 Agnostic Proper Learning and Two-Sided Sample-Based Testing

Write $\mathcal{C}(L)$ for the class of convex, L-Lipschitz functions:

$$\mathcal{C}(L) := \{f : \mathbb{R}^n \to \mathbb{R} : f \text{ is } L\text{-Lipschitz and convex}\}.$$

Recall our notation for the L_2 distance to the set of convex L-Lipschitz functions, which we may now write as

$$\mathrm{d}^L_{\mathrm{conv}}(f) = \inf_{g \in \mathcal{C}(L)} \|f - g\|_{L^2(\gamma)}.$$

By convention, given a function $f : \mathbb{R}^n \to \mathbb{R}$, by "sample from f" we mean a sample of the form $(\boldsymbol{x}, f(\boldsymbol{x}))$ where $\boldsymbol{x} \sim N(0, I_n)$. The main result in this section is the following.

▶ **Theorem 1** (Agnostic proper learning of Lipschitz convex functions). *Let $\varepsilon, L > 0$ and suppose $f : \mathbb{R}^n \to \mathbb{R}$ is a L-Lipschitz function. There exists an algorithm which, given i.i.d. access to labeled samples $(\boldsymbol{x}, f(\boldsymbol{x}))$ where $\boldsymbol{x} \sim N(0, I_n)$, draws $n^{O(L^2/\varepsilon^2)}$ samples, runs in time $\exp(\widetilde{O}(nL^2/\varepsilon^2))$, and outputs a function $g \in L^2(\gamma)$ such that*

$$\|f - g\|_{L^2(\gamma)} \le \mathrm{d}^L_{\mathrm{conv}}(f) + \varepsilon.$$

Additionally, the function g is convex and L-Lipschitz.

Combining the agnostic proper learning algorithm with a distance estimation step, we obtain a sample-based tolerant tester for Lipschitz convex functions:

▶ **Theorem 2** (Tolerant two-sided testing of Lipschitz convex functions). *Let $\varepsilon, \varepsilon_0, L \ge 0$ and suppose $f : \mathbb{R}^n \to \mathbb{R}$ is L-Lipschitz. There is an algorithm which, given i.i.d. labeled samples $(\boldsymbol{x}, f(\boldsymbol{x}))$ where $\boldsymbol{x} \sim N(0, I_n)$, draws $n^{O(L^2/\varepsilon^2)}$ samples, runs in time $\exp(\widetilde{O}(nL^2/\varepsilon^2))$, and has the following performance guarantee:*

- *If $\mathrm{d}^L_{\mathrm{conv}}(f) \le \varepsilon_0$, then it outputs "accept" with probability $9/10$;*
- *If $\mathrm{d}^L_{\mathrm{conv}}(f) \ge \varepsilon + \varepsilon_0$, then it outputs "reject" with probability $9/10$.*

Proof. Note that

$$\mathrm{d}^L_{\mathrm{conv}}(f)^2 \le \mathbf{Var}[f] \le \mathbf{E}\left[\|\nabla f\|^2\right] \le L^2$$

by the Poincaré inequality (Proposition 10), the fact that f is L-Lipschitz, and the fact that constant functions are convex. Hence we may assume that $\varepsilon \le L$, since otherwise the tester can immediately accept.

On input $f : \mathbb{R}^n \to \mathbb{R}$, the tester uses the learning algorithm from Theorem 1 to learn a function $g \in \mathcal{C}(L)$ satisfying $\|f - g\|_2 \le \mathrm{d}^L_{\mathrm{conv}}(f) + \varepsilon/4$ with probability at least $2/3$; assume henceforth that this event occurs. It then takes $m = O(L^4/\varepsilon^4)$ samples $(\boldsymbol{x}_i, f(\boldsymbol{x}_i))$, and accepts if and only if

$$\boldsymbol{Y} := \frac{1}{m}\sum_{i=1}^{m}(f(\boldsymbol{x}_i) - g(\boldsymbol{x}_i))^2 \le \left(\varepsilon_0 + \frac{\varepsilon}{2}\right)^2.$$

The sample and time complexity claims are immediate, and we now prove correctness. Let $h := f - g$, so that $\mathbf{E}[\boldsymbol{Y}] = \mathbf{E}[h^2] = \|f - g\|^2_{L^2(\gamma)}$ and $\mathbf{Var}[\boldsymbol{Y}] = \frac{1}{m}\mathbf{Var}[h^2]$. Define the centered function $\bar{h} := h - \mathbf{E}[h]$. Then $\bar{h}$ is centered and $2L$-Lipschitz, and hence sub-Gaussian with sub-Gaussian norm $O(L)$. Then

$$\mathbf{Var}[h^2] = \mathbf{Var}[(\bar{h} + \mathbf{E}[h])^2] = \mathbf{Var}[\bar{h}^2 + 2\,\mathbf{E}[h]\bar{h}] \le 2\,\mathbf{Var}[\bar{h}^2] + 2 \cdot \mathbf{E}[h]^2\,\mathbf{Var}[\bar{h}].$$

By the sub-Gaussianity of $\bar{h}$, we have $\mathbf{Var}[\bar{h}^2] = O(L^4)$ and $\mathbf{Var}[\bar{h}] = O(L^2)$. Moreover, we have

$$\mathbf{E}[h]^2 \le \mathbf{E}[h^2] = \|f-g\|_{L^2(\gamma)}^2 \le \left(\mathrm{d}_{\mathrm{conv}}^L(f) + \frac{\varepsilon}{4}\right)^2 \le 2\mathrm{d}_{\mathrm{conv}}^L(f)^2 + \varepsilon^2 \le O(L^2)\,.$$

We conclude that $\mathbf{Var}[h^2] \le O(L^4)$. By Chebyshev's inequality,

$$\mathbf{Pr}\left[\left|\boldsymbol{Y} - \|f-g\|_{L^2(\gamma)}^2\right| \ge \frac{\varepsilon^2}{16}\right] \le \frac{\frac{1}{m}\cdot O(L^4)}{\varepsilon^4/256} \le \frac{1}{10}\,,$$

for suitable choice of $m = O(L^4/\varepsilon^4)$. Suppose $\left|\boldsymbol{Y} - \|f-g\|_{L^2(\gamma)}^2\right| \le \varepsilon^2/16$. Then if $\mathrm{d}_{\mathrm{conv}}^L(f) \le \varepsilon_0$, we obtain

$$\boldsymbol{Y} \le \|f-g\|_{L^2(\gamma)}^2 + \frac{\varepsilon^2}{16} \le \left(\|f-g\|_{L^2(\gamma)} + \frac{\varepsilon}{4}\right)^2 \le \left(\mathrm{d}_{\mathrm{conv}}^L(f) + \frac{\varepsilon}{2}\right)^2 \le \left(\varepsilon_0 + \frac{\varepsilon}{2}\right)^2,$$

and the tester accepts. On the other hand, suppose $\mathrm{d}_{\mathrm{conv}}^L(f) > \varepsilon_0 + \varepsilon$. Then, recalling that g is a convex, L-Lipschitz function, we obtain

$$\begin{aligned}\boldsymbol{Y} &\ge \|f-g\|_{L^2(\gamma)}^2 - \frac{\varepsilon^2}{16} > (\varepsilon_0+\varepsilon)^2 - \frac{\varepsilon^2}{16} = \varepsilon_0^2 + 2\varepsilon_0\varepsilon + \frac{15\varepsilon^2}{16} \\ &> \varepsilon_0^2 + 2\varepsilon_0\cdot\frac{3\varepsilon}{4} + \left(\frac{3\varepsilon}{4}\right)^2 = \left(\varepsilon_0 + \frac{3\varepsilon}{4}\right)^2,\end{aligned}$$

and the tester rejects. By a union bound, the tester correctly accepts/rejects except with probability at most $\frac{2}{3} + \frac{1}{10} < \frac{1}{2}$, which may be made as small as desired by a standard majority vote. ◀

Proof of the Main Result

We require the following lemmas, whose proofs we defer to the full version. The first lemma says that a Lipschitz function is well-approximated by its low-degree Hermite expansion. This fact has been observed before, e.g. in [2, 46] for domain $[0,1]^n$; here we give a Gaussian version. Recall the notation $f^{\le d} := \sum_{|\alpha|\le d}\widehat{f}(\alpha)h_\alpha$ for the degree-d Hermite expansion of f.

▶ **Lemma 11** (Lipschitz functions are approximately low-degree). *Let $L, \varepsilon > 0$, let $f : \mathbb{R}^n \to \mathbb{R}$ be L-Lipschitz, and let $d := \lceil L^2/\varepsilon^2 \rceil$. Then $\|f - f^{\le d}\|_{L^2(\gamma)} \le \varepsilon$.*

Additionally, we now give upper bounds on the number of samples required to approximately learn the low-degree Hermite coefficients of a Lipschitz function. We assume for simplicity that f has expectation bounded by some constant (this assumption is not essential, and the general case is recovered in the proof of Theorem 1).

▶ **Lemma 12** (Learning a single Hermite coefficient). *Let $L, \xi > 0$, $\delta \in (0,1)$, $\alpha \in \mathbb{N}^n$, and let $f : \mathbb{R}^n \to \mathbb{R}$ be L-Lipschitz and satisfy $|\mathbf{E}[f]| \le O(1)$. Then*

$$O\left(\frac{L^2 3^{|\alpha|}}{\xi^2}\log\left(\frac{1}{\delta}\right)\right) \text{ samples from } f$$

suffice to compute an estimate $\widetilde{\boldsymbol{f}}(\alpha)$ satisfying $\left|\widetilde{\boldsymbol{f}}(\alpha) - \widehat{f}(\alpha)\right| \le \xi$ with probability at least $1-\delta$. The running time is polynomial in the number of samples.

We have the following immediate consequence:

▶ **Corollary 13** (Learning a low-degree expansion). *Let $L, \varepsilon > 0$, $\delta \in (0,1)$, $d \in \mathbb{N}$, and let $f : \mathbb{R}^n \to \mathbb{R}$ be L-Lipschitz and satisfy $|\mathbf{E}[f]| \le O(1)$. Then*

$$O\left(\frac{L^2 n^{2.01d}}{\varepsilon^2} \log\left(\frac{1}{\delta}\right)\right) \text{ samples from } f$$

suffice to obtain coefficient estimates $\widetilde{\boldsymbol{f}}(\alpha)$, for all $\alpha \in \mathbb{N}^n$ with $|\alpha| \le d$, such that the function $\boldsymbol{f} := \sum_{|\alpha| \le d} \widetilde{\boldsymbol{f}}(\alpha) h_\alpha$ satisfies $\|f^{\le d} - \boldsymbol{f}\|_{L^2(\gamma)} \le \varepsilon$ with probability at least $1 - \delta$. The running time is polynomial in the number of samples.

We will require a simple a priori upper bound on the Lipschitz constant of the low-degree approximation above in terms of distance to the origin:

▶ **Lemma 14** (Lipschitz constant of low-degree polynomials). *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a degree-d polynomial. Then for all $x \in \mathbb{R}^n$, we have*

$$\|\nabla f(x)\|_2 \le \|f\|_{L^2(\gamma)} d^{d+\frac{1}{2}} n^{d/2} (1 + |x|)^d .$$

The next lemma says that, given *query* access to a Lipschitz function g which is close to some convex Lipschitz function over a ball, we may learn (via convex regression) a good convex approximation to g using a finite number of queries. We will apply this result to an approximation $\boldsymbol{f}$ of the low-degree restriction $f^{\le d}$ over the restriction of the Gaussian measure to a sufficiently large ball. It is convenient to work with the centered radius-r ℓ^∞-ball $B_\infty(r)$, i.e. an axis-aligned hypercube.

▶ Remark 15. This number of queries used for the convex regression is exponential in n; however, we make these queries to the learned approximation $\boldsymbol{f}$ rather than the input function f, so we pay this cost in the running time but not in the sample complexity.

▶ **Lemma 16** (Learning a Lipschitz convex approximation). *Let $r > 0$, and let μ be the probability measure γ conditioned to $B_\infty(r)$. There exists an algorithm which, given parameters $L, L', \varepsilon > 0$, with $L' \ge L$, and query access to some L'-Lipschitz function $g : B_\infty(r) \to \mathbb{R}$, makes $(L' r \sqrt{n}/\varepsilon)^{O(n)}$ queries to g and outputs a function $\widetilde{g} \in \mathcal{C}(L)$ that is a nearly optimal convex approximation of g over $B_\infty(r)$ with respect to L-Lipschitz functions, in the following sense: for all convex, L-Lipschitz $h : B_\infty(r) \to \mathbb{R}$, we have*

$$\|\widetilde{g} - g\|_{L^2(B_\infty(r),\mu)} \le \|h - g\|_{L^2(B_\infty(r),\mu)} + \varepsilon.$$

The running time is polynomial in the number of queries.

To translate back and forth between full Gaussian space and its restriction to an ℓ^∞-ball, while approximately preserving the distance between Lipschitz functions, we use the following lemma.

▶ **Lemma 17** (Restricting Gaussian space to a box). *There exists a universal constant $C > 0$ such that the following holds. Let $L, \tau \ge 1$, $\varepsilon \in (0, L)$, and let $r := C\sqrt{\log(nL\tau/\varepsilon)}$. Let μ be probability measure γ conditioned to $B_\infty(r)$. Then for all L-Lipschitz functions $f, g : \mathbb{R}^n \to \mathbb{R}$ such that $|f(x^*) - g(x^*)| \le \tau$ for some $x^* \in B_\infty(r)$, we have*

$$\left| \|f - g\|_{L^2(\gamma)} - \|f - g\|_{L^2(B_\infty(r),\mu)} \right| \le \varepsilon .$$

▶ Remark 18. The condition that $|f(x) - g(x)| \le \tau$ for some $x \in B_\infty(r)$ in Lemma 17 holds in particular if $\|f - g\|_2 \le \tau/2$, since otherwise the choice of radius r is large enough to yield (say) $\|f - g\|_2 \ge 3\tau/4$, a contradiction.

Equipped with these results, we may prove Theorem 1.

Proof of Theorem 1. We first claim that we may assume without loss of generality that $|\mathbf{E}[f]| \le \varepsilon/100$. Indeed, the algorithm may first compute an approximation $\bar{\boldsymbol{f}}$ of $\mathbf{E}[f]$ up to $\varepsilon/100$ additive error using $O(L^2/\varepsilon^2)$ samples (by Chebyshev's inequality, since $\mathbf{Var}[f] \le \mathbf{E}[\|\nabla f\|_2^2] \le L^2$ by Proposition 10, and the Lipschitz assumption), then proceed with the modified input function $f - \bar{\boldsymbol{f}}$ and shift the final output again by $\bar{\boldsymbol{f}}$.

We may also assume that $L \ge 1$ (by rescaling the input and ε if necessary) and that $\varepsilon < \frac{100}{99}L$ (otherwise, we get $\|f\|_{L^2(\gamma)} \le \|f - \mathbf{E}[f]\|_{L^2(\gamma)} + |\mathbf{E}[f]| \le L + \varepsilon/100 \le \varepsilon$ by the triangle and Poincaré inequalities, and we may immediately output the constant zero function).

Let $\varepsilon' := \varepsilon/8$ and $d := \lceil (L/\varepsilon')^2 \rceil$. The algorithm first uses

$$O\left(\frac{L^2 n^{2.01d}}{\varepsilon^2}\right) = n^{O(L^2/\varepsilon^2)} \text{ samples from } f$$

to learn, via Corollary 13, a degree-d approximation $\boldsymbol{f}$ satisfying $\|f^{\le d} - \boldsymbol{f}\|_{L^2(\gamma)} \le \varepsilon'$ with probability at least $2/3$; assume henceforth that this event occurs. By Lemma 11 and the triangle inequality, we have $\|f - \boldsymbol{f}\|_{L^2(\gamma)} \le 2\varepsilon'$. Note that $\boldsymbol{f}$ satisfies $\|\boldsymbol{f}\|_{L^2(\gamma)} \le \|f\|_{L^2(\gamma)} + 2\varepsilon' \le L + \varepsilon/100 + 2\varepsilon' \le 2L$.

Let $r = \Theta\left(\frac{L}{\varepsilon}\sqrt{\log(nL/\varepsilon)}\right)$ be large enough to ensure that, when we set

$$L' := 2L(2ndr)^{2d} \ge \|\boldsymbol{f}\|_{L^2(\gamma)}(2ndr)^{2d} \ge \|\boldsymbol{f}\|_{L^2(\gamma)} d^{d+\frac{1}{2}} n^{d/2}(1 + r\sqrt{n})^d$$

as the upper bound on the Lipschitz constant of $\boldsymbol{f}$ over $B_\infty(r)$ via Lemma 14, we obtain

$$r \ge C'\sqrt{\log(nL'/\varepsilon)}$$

for sufficiently large choice of absolute constant $C' > 0$ so that, by Lemma 17, ball $B_\infty(r)$ gives an ε'-approximation for the distance between $\boldsymbol{f}$ and L-Lipschitz functions g for parameter $\tau = 6L$. (One may check that it is possible to choose r satisfying this.)

Let $\widetilde{g} \in \mathcal{C}(L)$ be the function produced by Lemma 16 with input $g = \boldsymbol{f}$, the choice of r above, and parameters L, L' and ε'. Note that the number of queries to $\boldsymbol{f}$, and thus the running time of this step, is

$$(L'r\sqrt{n}/\varepsilon)^{O(n)} = (Lndr/\varepsilon)^{O(dn)} = (nL/\varepsilon)^{O(nL^2/\varepsilon^2)} = \exp(\widetilde{O}(nL^2/\varepsilon^2))\,.$$

We claim that $\widetilde{g}$ satisfies the guarantee of Theorem 1. We already have that $\widetilde{g} \in \mathcal{C}(L)$, so it remains to show that $\|\widetilde{g} - f\|_{L^2(\gamma)} \le \mathrm{d}^L_{\mathrm{conv}}(f) + \varepsilon$. Let $h \in \mathcal{C}(L)$ be any function satisfying $\|f - h\|_{L^2(\gamma)} \le L$ (which exists because f is L-close to the constant $|\mathbf{E}[f]|$ function by the Poincaré inequality), so that $\|\boldsymbol{f} - h\|_{L^2(\gamma)} \le L + 2\varepsilon' \le 2L$. We have

$$\begin{aligned}
\|\widetilde{g} - f\|_{L^2(\gamma)} &\le \|\widetilde{g} - \boldsymbol{f}\|_{L^2(\gamma)} + \|\boldsymbol{f} - f\|_{L^2(\gamma)} \\
&\le \|\widetilde{g} - \boldsymbol{f}\|_{L^2(B_\infty(r),\mu)} + 3\varepsilon' && \text{(Lemma 17, see below)} \\
&\le \|h - \boldsymbol{f}\|_{L^2(B_\infty(r),\mu)} + 4\varepsilon' && \text{(Lemma 16)} \\
&\le \|h - \boldsymbol{f}\|_{L^2(\gamma)} + 5\varepsilon' && \text{(Lemma 17 via Remark 18)} \\
&\le \|h - f\|_{L^2(\gamma)} + \|f - \boldsymbol{f}\|_{L^2(\gamma)} + 5\varepsilon' \\
&\le \|h - f\|_{L^2(\gamma)} + 7\varepsilon'\,.
\end{aligned}$$

To justify the first use of Lemma 17, we claim that $\widetilde{g}$ satisfies $|\widetilde{g}(x) - \boldsymbol{f}(x)| \leq 6L = \tau$ at some point $x \in B_\infty(r)$. Indeed, note that the function h satisfies (by Lemma 17 via Remark 18) $\|h - \boldsymbol{f}\|_{L^2(B_\infty(r),\mu)} \leq \|h - \boldsymbol{f}\|_{L^2(\gamma)} + \varepsilon' \leq 3L$; hence, if we had $|\widetilde{g}(x) - \boldsymbol{f}(x)| > 6L$ for all $x \in B_\infty(r)$, we would obtain $\|\widetilde{g} - \boldsymbol{f}\|_{L^2(B_\infty(r),\mu)} > 6L > \|h - \boldsymbol{f}\|_{L^2(B_\infty(r),\mu)} + \varepsilon'$, contradicting Lemma 16.

Since the above holds for all $h \in \mathcal{C}(L)$ sufficiently close to f, we obtain $\|\widetilde{g} - f\|_{L^2(\gamma)} \leq \mathrm{d}^L_{\mathrm{conv}}(f) + \varepsilon$ as desired. ◀

4 One-Sided Sample-Based Testing

We now turn to convexity testing algorithms with *one-sided* error, i.e., algorithms that never incorrectly reject a convex function. Our main result here is the following:

▶ **Theorem 4** (One-sided testing of Lipschitz convex functions). *Let $f : \mathbb{R}^n \to \mathbb{R}$ be L-Lipschitz, and let $\varepsilon \geq 0$. There is an algorithm, One-Sided-Tester (Algorithm 1), which, given access to i.i.d. labeled samples $(\boldsymbol{x}, f(\boldsymbol{x}))$ where $\boldsymbol{x} \sim N(0, I_n)$, makes $O(\sqrt{n}L/\varepsilon)^n$ draws, runs in time $(nL/\varepsilon)^{O(n)}$, and has the following performance guarantee:*

- *If f is convex, then One-Sided-Tester outputs "accept" with probability 1;*
- *If $\mathrm{d}^L_{\mathrm{conv}}(f) \geq \varepsilon$, then One-Sided-Tester outputs "reject" with probability 9/10.*

Note that in contrast to our upper bound for standard testing (Theorem 2), our sample complexity is exponentially worse.

Our one-sided testing algorithm proceeds by constructing an empirical *convex envelope* of the function using labeled samples. We will ensure by design that when f is convex, its empirical convex envelope is close (in an $L^\infty(\gamma)$ sense) to f. On the other hand, when f is far from convex, it will be far from its empirical convex envelope, which will be revealed by the samples. The strong $L^\infty(\gamma)$ guarantee of the convex envelope may be contrasted with our two-sided tester from Theorem 2, which also proceeds via learning but rather in $L^2(\gamma)$ sense (which is a weaker guarantee that more precisely matches our target metric for testing).

4.1 The Empirical Convex Envelope

We will require the following definition:

▶ **Definition 19** (Empirical convex envelope). *For a function $f : \mathbb{R}^n \to \mathbb{R}$ and m points $Y = \{y_1, y_2, \ldots, y_m\} \subset \mathbb{R}^n$, we define the* empirical convex envelope *of f with respect to Y as the function $\overline{f_Y} : \mathbb{R}^n \to \mathbb{R}$ defined by*

$$\overline{f_Y}(x) := \sup\left\{\langle p, x\rangle + a : p \in \mathbb{R}^n, a \in \mathbb{R}, \|p\|_2 \leq L, \langle p, y\rangle + a \leq f(y) \text{ for all } y \in Y\right\}.$$

Informally, $\overline{f_Y}$ is the "largest" convex function that is below f on all the sampled points $y \in Y$. We record two immediate properties of the empirical convex envelope for future use:

▶ **Lemma 20.** *Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is L-Lipschitz, and let $Y = \{y_1, \ldots, y_m\} \subseteq \mathbb{R}^n$. Let $\overline{f_Y} : \mathbb{R}^n \to \mathbb{R}$ be the empirical convex envelope of f with respect to Y.*

1. *The empirical convex envelope $\overline{f_Y}$ is convex and L-Lipschitz.*
2. *Furthermore, if f is itself convex, then for all $y \in Y$, $\overline{f_Y}(y) = f(y)$.*

Proof. Note that convexity of $\overline{f_Y}$ is immediate from Definition 19 since the supremum of affine functions is convex. Additionally, the constraint that $\|p\|_2 \leq L$ ensures that $\overline{f_Y}$ is L-Lipschitz. Together, these imply the first item.

We now turn to the second item. Since $\overline{f_Y}(y_j) \leq f(y_j)$ for all $y_j \in Y$ by construction, it remains to show that $f(y_j) \leq \overline{f_Y}(y_j)$ for $y_j \in Y$. This is readily seen to hold by taking $p = \nabla f(y_j)$ and $a = f(y_j)$. Since f is L-Lipschitz, and by Rademacher's theorem Lipschitz functions are differentiable almost everywhere, it follows that $\|\nabla f(y_j)\|_2 \leq L$. Since $\overline{f_Y}(y_j)$ is a supremum over all feasible p and a, it follows that $\overline{f_Y}(y_j) \geq f(y_j)$. ◀

Next, we note that the empirical convex envelope $\overline{f_Y}$ can be computed on an input $x \in \mathbb{R}^n$ by the quadratically constrained convex program CECE (short for "compute empirical convex envelope") given in Figure 2.

▶ **Lemma 21.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be L-Lipschitz, and let $Y = \{y_1, \ldots, y_m\} \subseteq \mathbb{R}^n$. Let $\overline{f_Y} : \mathbb{R}^n \to \mathbb{R}$ be the empirical convex envelope of f with respect to Y. Then for each $x \in \mathbb{R}^n$, the program CECE(x) is feasible and satisfies CECE$(x) = \overline{f_Y}(x)$. Additionally, CECE(x) runs in time* $\mathrm{poly}(m, n)$.

Here, we are assuming the real-valued model of computation, where infinite-length real-valued computations can be achieved with one unit operation. However, our results work equally well in the bounded-bit model, where we use and assume finite bits of precision, with some minor (and routine) modifications to the analysis.

Proof. Note that the quadratic program in Figure 2 coincides with the definition of the empirical convex envelope from Definition 19, and so correctness holds. (Feasibility can be verified by taking $p = 0$ and a small enough, since the set Y is finite.) The runtime is also immediate via standard guarantees for solving convex programs [15]. ◀

$$
\begin{array}{ll}
\text{maximize} & \langle p, x \rangle + a \\
\text{subject to} & p \in \mathbb{R}^n, a \in \mathbb{R} \\
& \|p\|_2 \leq L \\
& \langle p, y \rangle + a \leq f(y) \quad \forall y \in Y
\end{array}
$$

Figure 2 The quadratic program CECE(x) which takes as input a collection of samples $Y = \{y_1, \ldots, y_m\}$ labeled by an L-Lipschitz function f.

4.2 Algorithm 1 and Technical Results

We now formally describe our one-sided convexity testing algorithm, One-Sided-Tester(cf. Algorithm 1). We also record some useful lemmas before proving Theorem 4. For the remainder of this section, we set $d := \frac{\varepsilon}{8L}$, where ε, L are as in Theorem 4.

The first lemma shows that the distance from f to its convex envelope $\overline{f_Y}$ can be well-approximated by considering only the points in the set Y.

▶ **Lemma 22.** *Let $Y = \{y_1, y_2, \ldots, y_m\}$ be a set of points in $\mathbb{R}^n$. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a L-Lipschitz function and let $\overline{f_Y}$ be its empirical convex envelope with respect to Y. Then for all $x \in \mathbb{R}^n$, we have*

$$|\overline{f_Y}(x) - f(x)| \leq \min_{y \in Y} \left\{ \left|\overline{f_Y}(y) - f(y)\right| + 2L\|x - y\|_2 \right\}.$$

Algorithm 1 A one-sided testing algorithm for Lipschitz convex functions.

Input: Access to i.i.d. labeled samples $(\boldsymbol{x}, f(\boldsymbol{x}))$ where f is L-Lipschitz, $\varepsilon \geq 0$

Output: "Accept" or "reject"

One-Sided-Tester:

1. Set
$$t_1 := \left(\frac{cL\sqrt{n}}{\varepsilon}\right)^n$$
for a universal constant c, and draw t_1 samples $(\boldsymbol{x}^{(1)}, f(\boldsymbol{x}^{(1)})), \ldots, (\boldsymbol{x}^{(t_1)}, f(\boldsymbol{x}^{(t_1)}))$. Let $\boldsymbol{Y} := \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(t_1)}\}$.
2. Let $\mathcal{P}$ be a partition of $B_2(1.1\sqrt{n})$ into at most $(3nL/\varepsilon)^n$ (solid) hypercubes of side length $\ell = \lceil \varepsilon/(4L\sqrt{n}) \rceil$. For each hypercube $C \in \mathcal{P}$, check if there exists $\boldsymbol{x}^{(i)} \in C$; if not, halt and return "accept."
3. If $f(\boldsymbol{x}^{(j)}) \neq \mathrm{CECE}(\boldsymbol{x}^{(j)})$ for any $j \in [t_1]$, then halt and return "reject." Otherwise, return "accept."

Proof. Fix $x \in \mathbb{R}^n$. For each $y \in Y$, we have

$$\begin{aligned} |\overline{f_Y}(x) - f(x)| &\leq |\overline{f_Y}(x) - \overline{f_Y}(y)| + |\overline{f_Y}(y) - f(y)| + |f(x) - f(y)| && \text{(triangle inequality)} \\ &\leq |\overline{f_Y}(y) - f(y)| + 2L \cdot \|x - y\|_2 && (f, \overline{f_Y} \text{ are } L\text{-Lipschitz}) \end{aligned}$$

as desired. ◀

The following lemma shows that with high probability, a draw of roughly $\exp(n \log n)$ samples from $N(0, I_n)$ will be sufficiently "dense" in $B_2(1.1\sqrt{n})$, which will help us bound the error term in Lemma 22. The proof computes a lower bound of the Gaussian density inside $B_2(1.1\sqrt{n})$, and then uses a union bound to show that, with high constant probability, each small cube in a partition of $B_2(1.1\sqrt{n})$ contains at least one sample.

▶ **Lemma 23.** *Let t_1 be as in Algorithm 1, and suppose $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(t_1)} \sim N(0, I_n)$ are i.i.d. draws. Then with probability 0.99, for all $x \in B_2(1.1\sqrt{n})$ we have*

$$\min_{i \in [t_1]} \|x - \boldsymbol{x}^{(i)}\| \leq d\,.$$

Proof. Note that $B_2(1.1\sqrt{n}) \subseteq B_\infty(1.1\sqrt{n})$. Partition $B_\infty(1.1\sqrt{n})$ into a hypergrid of at most $(1.1\sqrt{n}\ell^{-1})^n \leq \left(3nd^{-1}\right)^n$ (hyper)cubes, each of side length $\ell := \lceil d/\sqrt{n} \rceil$. Suppose for every cube in the grid there exists a point $y \in Y$ in the cube. Then note that

$$\min_{y \in Y} \|x - y\|_2 \leq \ell \cdot \sqrt{n} \leq d$$

thanks to our choice of ℓ.

Thus, in order to establish Lemma 23, it suffices to show that after drawing t_1 samples, with probability at least 9/10, every cube in the hypergrid contains at least one sample. We will show this by taking a union bound over the $(3nd^{-1})^n$ cubes in the hypergrid.

In this argument, it will be necessary to consider the cube in the hypergrid with minimal Gaussian measure. Note that the maximum ℓ_2 norm of any point in the hypergrid is at most $(1.1 + \ell)\sqrt{n}$. Therefore for all $x \in B_\infty(1.1\sqrt{n})$, we have

$$\frac{\exp\left(-||x||_2^2/2\right)}{(2\pi)^{n/2}} \geq \frac{\exp\left(-(1.1+\ell)^2/(2n)\right)}{(2\pi)^{n/2}} = \frac{1}{(2\pi)^{n/2}} \cdot \exp\left(-\left(1.1 + \frac{\varepsilon}{(4L\sqrt{n})}\right)^2 \cdot \frac{1}{2n}\right).$$

Since the Lebesgue measure of any cube is ℓ^n, the minimum Gaussian measure of any of the cubes in our hypergrid is at least

$$\alpha := \frac{1}{(2\pi)^{n/2}} \cdot \exp\left(-\left(1.1 + \frac{\varepsilon}{(4L\sqrt{n})}\right)^2 \cdot \frac{1}{2n}\right) \cdot \ell^n \,.$$

The probability that t_1 independent draws from $N(0, I_n)$ all fail to land in this cube is

$$(1-\alpha)^{t_1} \leq \exp(-\alpha t_1) \,.$$

Applying a union bound, it follows that the probability that there exists a cube in the hypergrid not containing any sample is at most

$$\exp(-\alpha t_1) \cdot 2^{O(n \log(1.1n/d)} \,,$$

and it is readily verified that this is at most 0.01 for our choice of t_1 and ℓ, for some universal constant c. ◀

Finally, the next lemma shows that, under the density condition above, if f and $\overline{f_Y}$ agree on all points in Y then they are close in $L^2(\gamma)$.

▶ **Lemma 24.** *Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is an L-Lipschiz function and $Y = \{y_1, y_2, \ldots, y_m\}$ is a set of points in $\mathbb{R}^n$ such that for all $x \in B(1.1\sqrt{n})$, there exists $i \in [m]$ such that $\|x - y_i\|_2 \leq d$, for $d = \varepsilon/(8L)$. Let $\overline{f_Y}$ be the empirical convex envelope of f with respect to Y. If $f(y_i) = \overline{f_Y}(y_i)$ for all $i \in [m]$, then $\left\|f - \overline{f_Y}\right\|_{L^2(\gamma)} \leq \varepsilon/2$.*

Proof. We decompose $\left\|f - \overline{f_Y}\right\|^2_{L^2(\gamma)}$ into the regions inside and outside $B_2(1.1\sqrt{n})$ and apply Lemma 22:

$$\begin{aligned}
&\left\|f - \overline{f_Y}\right\|^2_{L^2(\gamma)} \\
&= \mathop{\mathbf{E}}_{\boldsymbol{x}\sim N(0,I_n)}\left[\left|f(\boldsymbol{x}) - \overline{f_Y}(\boldsymbol{x})\right|^2\right] \\
&\leq \mathop{\mathbf{E}}_{\boldsymbol{x}\sim N(0,I_n)}\left[\left|f(\boldsymbol{x}) - \overline{f_Y}(\boldsymbol{x})\right|^2 \,\middle|\, \|\boldsymbol{x}\|_2 \leq 1.1\sqrt{n}\right] \\
&\quad + \mathop{\mathbf{E}}_{\boldsymbol{x}\sim N(0,I_n)}\left[\left|f(\boldsymbol{x}) - \overline{f_Y}(\boldsymbol{x})\right|^2 \,\middle|\, \|\boldsymbol{x}\|_2 > 1.1\sqrt{n}\right] \cdot \mathbf{Pr}\left[\|\boldsymbol{x}\|_2 > 1.1\sqrt{n}\right] \\
&\leq (2Ld)^2 + (2L)^2 \mathop{\mathbf{E}}_{\boldsymbol{x}\sim N(0,I_n)}\left[\min_{y\in Y} \|\boldsymbol{x} - y\|_2^2 \mid \|\boldsymbol{x}\|_2 > 1.1\sqrt{n}\right] \cdot \mathop{\mathbf{Pr}}_{\boldsymbol{x}\sim N(0,I_n)}\left[\|\boldsymbol{x}\|_2 > 1.1\sqrt{n}\right] \\
&= (2Ld)^2 + (2L)^2 \cdot \mathrm{poly}(n)\exp(-\Omega(n)) \\
&= \varepsilon^2/16 + o(1) \,,
\end{aligned}$$

which is less than $\varepsilon^2/4$ for large enough n. ◀

4.3 Proof of Theorem 4

Proof. First, note that the sample complexity of One-Sided-Tester is $O\left(\sqrt{n}L/\varepsilon\right)^n$ as desired. We establish correctness as well as the runtime bound below, starting with the former.

Correctness. We first show that if f is L-Lipschitz and convex, then One-Sided-Tester accepts with probability 1. Indeed, Algorithm 1 only outputs "reject" when $f(y) \neq \mathrm{CECE}(y)$ for some $y \in Y$, which cannot happen by Lemmas 20 and 21.

For soundness, suppose that $\mathrm{d}^L_{\mathrm{conv}}(f) \geq \varepsilon$, so that in particular $\left\|f - \overline{f_Y}\right\|_{L^2(\gamma)} \geq \varepsilon$. Suppose the algorithm samples a set Y of points in $\mathbb{R}^n$ such that for all $x \in B_2(1.1\sqrt{n})$, there exists $y \in Y$ such that $\|x - y\|_2 \leq d$, for $d = \varepsilon/(8L)$; by Lemma 23 this occurs with probability at least 0.99. In this case the algorithm rejects, because it would only accept if we had $f(y) = \mathrm{CECE}(y)$ for every $y \in Y$; since $\mathrm{CECE}(y) = \overline{f_Y}(y)$ for each $y \in Y$ by Lemma 21, this would imply via Lemma 24 that $\left\|f - \overline{f_Y}\right\|_{L^2(\gamma)} \leq \varepsilon/2$, a contradiction. Therefore the algorithm rejects with probability at least 0.99.

Runtime. Note that Step 1 takes time t_1 and Step 2 takes time

$$|\mathcal{P}| \cdot t_1 = (3nL/\varepsilon)^n \cdot \left(\frac{50L\sqrt{n}}{\varepsilon}\right)^n .$$

Finally, Step 3 requires time $\mathrm{poly}(n, t_1)$ by Lemma 21. It follows that Algorithm 1 uses time $(nL/\varepsilon)^{O(n)}$. ◀

References

1. Amir Ali Ahmadi, Alex Olshevsky, Pablo A Parrilo, and John N Tsitsiklis. NP-hardness of deciding convexity of quartic polynomials and related problems. *Mathematical programming*, 137:453–476, 2013. doi:10.1007/s10107-011-0499-2.
2. Alexandr Andoni, Rina Panigrahy, Gregory Valiant, and Li Zhang. Learning sparse polynomial functions. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 500–510. SIAM, 2014. doi:10.1137/1.9781611973402.37.
3. Dominique Bakry, Ivan Gentil, and Michel Ledoux. *Analysis and Geometry of Markov Diffusion Operators*. Grundlehren der mathematischen Wissenschaften. Springer Cham, 2014. doi:10.1007/978-3-319-00227-9.
4. Aleksandrs Belovs, Eric Blais, and Abhinav Bommireddi. Testing convexity of functions over finite domains. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2030–2045. SIAM, 2020. doi:10.1137/1.9781611975994.125.
5. Omri Ben-Eliezer. Testing Local Properties of Arrays. In *10th Innovations in Theoretical Computer Science Conference (ITCS 2019)*, Leibniz International Proceedings in Informatics (LIPIcs), pages 11:1–11:20, 2019. doi:10.4230/LIPIcs.ITCS.2019.11.
6. Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev. L_p-testing. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 164–173, 2014. doi:10.1145/2591796.2591887.
7. Arnab Bhattacharyya and Yuichi Yoshida. *Property Testing: Problems and Techniques*. Springer, 2022. doi:10.1007/978-981-16-8622-1.
8. Hadley Black. Nearly optimal bounds for sample-based testing and learning of k-monotone functions. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2024)*, pages 37–1. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.APPROX/RANDOM.2024.37.
9. Hadley Black, Eric Blais, and Nathaniel Harms. Testing and Learning Convex Sets in the Ternary Hypercube. In *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*, pages 15–1. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ITCS.2024.15.
10. Eric Blais and Abhinav Bommireddi. On testing and robust characterizations of convexity. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pages 18:1–18:15, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.18.

11 Eric Blais, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Lower bounds for testing properties of functions over hypergrid domains. In *2014 IEEE 29th Conference on Computational Complexity (CCC)*, pages 309–320. IEEE, 2014. doi:10.1109/CCC.2014.38.

12 Avrim Blum, Merrick L. Furst, Jeffrey C. Jackson, Michael J. Kearns, Yishay Mansour, and Steven Rudich. Weakly learning DNF and characterizing statistical query learning using fourier analysis. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing*, pages 253–262. ACM, 1994. doi:10.1145/195058.195147.

13 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 73–83, 1990. doi:10.1145/100216.100225.

14 Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT 2010*, pages 177–186. Physica-Verlag HD, 2010. doi:10.1007/978-3-7908-2604-3_16.

15 Stephen P Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. doi:10.1017/CBO9780511804441.

16 Nader H. Bshouty and Vitaly Feldman. On using extended statistical queries to avoid membership queries. *J. Mach. Learn. Res.*, 2:359–395, 2002. URL: https://jmlr.org/papers/v2/bshouty02a.html.

17 Nader H Bshouty and Christino Tamon. On the Fourier spectrum of monotone functions. *Journal of the ACM (JACM)*, 43(4):747–770, 1996. doi:10.1145/234533.234564.

18 Deeparnab Chakrabarty and Comandur Seshadhri. A $o(n)$ monotonicity tester for boolean functions over the hypercube. In *Proceedings of the forty-fifth annual ACM symposium on Theory of Computing*, pages 411–418, 2013. doi:10.1145/2488608.2488660.

19 Sitan Chen, Aravind Gollakota, Adam R. Klivans, and Raghu Meka. Hardness of noise-free learning for two-hidden-layer neural networks. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022*, 2022. URL: http://papers.nips.cc/paper_files/paper/2022/hash/45a7ca247462d9e465ee88c8a302ca70-Abstract-Conference.html.

20 X. Chen, A. Freilich, R. Servedio, and T. Sun. Sample-based high-dimensional convexity testing. In *Proceedings of the 17th Int. Workshop on Randomization and Computation (RANDOM)*, pages 37:1–37:20, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.37.

21 Xi Chen, Anindya De, Shivam Nadimpalli, Rocco A Servedio, and Erik Waingarten. Lower bounds for convexity testing. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 446–488. SIAM, 2025. doi:10.1137/1.9781611978322.13.

22 Anindya De, Elchanan Mossel, and Joe Neeman. Is your function low dimensional? In Alina Beygelzimer and Daniel Hsu, editors, *Conference on Learning Theory, COLT 2019*, volume 99 of *Proceedings of Machine Learning Research*, pages 979–993. PMLR, 2019. URL: http://proceedings.mlr.press/v99/de19a.html.

23 Anindya De, Elchanan Mossel, and Joe Neeman. Robust testing of low dimensional functions. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 584–597, 2021. doi:10.1145/3406325.3451115.

24 Anindya De, Shivam Nadimpalli, and Rocco A. Servedio. Quantitative correlation inequalities via semigroup interpolation. In *12th Innovations in Theoretical Computer Science Conference, ITCS 2021*, volume 185 of *LIPIcs*, pages 69:1–69:20, 2021. doi:10.4230/LIPIcs.ITCS.2021.69.

25 Anindya De, Shivam Nadimpalli, and Rocco A. Servedio. Convex influences. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS*, volume 215 of *LIPIcs*, pages 53:1–53:21, 2022. doi:10.4230/LIPIcs.ITCS.2022.53.

26 Anindya De, Shivam Nadimpalli, and Rocco A. Servedio. Testing Convex Truncation. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 4050–4082, 2023. doi:10.1137/1.9781611977554.ch155.

27 Anindya De, Shivam Nadimpalli, and Rocco A Servedio. Gaussian approximation of convex sets by intersections of halfspaces. In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1911–1930. IEEE, 2024. doi:10.1109/FOCS61266.2024.00115.

28 Anindya De and Rocco Servedio. Weak learning convex sets under normal distributions. In *Conference on Learning Theory*, pages 1399–1428. PMLR, 2021. URL: http://proceedings.mlr.press/v134/de21a.html.

29 Ronald A. DeVore and George G. Lorentz. *Constructive Approximation*, volume 303 of *Grundlehren der Mathematischen Wissenschaften*. Springer-Verlag, Berlin, 1993.

30 Ilias Diakonikolas, Daniel Kane, and Nikos Zarifis. Near-optimal SQ lower bounds for agnostically learning halfspaces and relus under gaussian marginals. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020. URL: https://proceedings.neurips.cc/paper/2020/hash/9d7311ba459f9e45ed746755a32dcd11-Abstract.html.

31 Ilias Diakonikolas, Daniel M Kane, Vasilis Kontonis, Christos Tzamos, and Nikos Zarifis. Agnostic proper learning of halfspaces under gaussian marginals. In *Conference on Learning Theory*, pages 1522–1551. PMLR, 2021. URL: http://proceedings.mlr.press/v134/diakonikolas21b.html.

32 Lutz Dümbgen. Shape-constrained statistical inference. *Annual Review of Statistics and Its Application*, 11, 2024. doi:10.1146/annurev-statistics-033021-014937.

33 Vitaly Feldman. A complete characterization of statistical query learning with applications to evolvability. *J. Comput. Syst. Sci.*, 78(5):1444–1459, 2012. doi:10.1016/j.jcss.2011.12.024.

34 Renato Ferreira Pinto Jr. Directed Isoperimetry and Monotonicity Testing: A Dynamical Approach. In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 2295–2305. IEEE, 2024. doi:10.1109/FOCS61266.2024.00134.

35 Surbhi Goel, Aravind Gollakota, and Adam R. Klivans. Statistical-query lower bounds via functional gradients. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020. URL: https://proceedings.neurips.cc/paper/2020/hash/17257e81a344982579af1ae6415a7b8c-Abstract.html.

36 Oded Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017. doi:10.1017/9781108135252.

37 Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998. doi:10.1145/285055.285060.

38 Oded Goldreich, Shafi Goldwassert, Eric Lehman, and Dana Ron. Testing monotonicity. In *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No. 98CB36280)*, pages 426–435. IEEE, 1998. doi:10.1109/SFCS.1998.743493.

39 Bram L Gorissen, İhsan Yanıkoğlu, and Dick Den Hertog. A practical guide to robust optimization. *Omega*, 53:124–137, 2015. doi:10.1016/j.omega.2014.12.006.

40 Piet Groeneboom and Geurt Jongbloed. *Nonparametric estimation under shape constraints*. Cambridge University Press, 2014. doi:10.1017/CBO9781139020893.

41 Adityanand Guntuboyina and Bodhisattva Sen. Nonparametric shape-restricted regression. *Statistical Science*, 33(4):568–594, 2018. doi:10.1214/18-STS665.

42 Anxin Guo and Aravindan Vijayaraghavan. Agnostic Learning of Arbitrary ReLU Activation under Gaussian Marginals. In *The Thirty Eighth Annual Conference on Learning Theory*, volume 291 of *Proceedings of Machine Learning Research*, pages 2592–2631. PMLR, 2025. URL: https://proceedings.mlr.press/v291/guo25a.html.

43 Prahladh Harsha, Adam Klivans, and Raghu Meka. An invariance principle for polytopes. *Journal of the ACM (JACM)*, 59(6):1–25, 2013. doi:10.1145/2395116.2395118.

44 Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2 edition, 2009. doi:10.1007/978-0-387-84858-7.

45 Aicke Hinrichs, Erich Novak, and Henryk Woźniakowski. The curse of dimensionality for the class of monotone functions and for the class of convex functions. *Journal of Approximation Theory*, 163(8):955–965, 2011. doi:10.1016/j.jat.2011.02.009.

46 Daniel Hsu, Clayton H. Sanford, Rocco A. Servedio, and Emmanouil Vasileios Vlatakis-Gkaragkounis. On the Approximation Power of Two-Layer Networks of Random ReLUs. In *Conference on Learning Theory, COLT 2021*, volume 134 of *Proceedings of Machine Learning Research*, pages 2423–2461. PMLR, 2021. URL: http://proceedings.mlr.press/v134/hsu21a.html.

47 Yaozhong Hu. Itô-wiener chaos expansion with exact residual and correlation, variance inequalities. *Journal of Theoretical Probability*, 10(4):835–848, 1997. doi:10.1023/A:1022654314791.

48 Michael J Kearns and Umesh Vazirani. *An introduction to computational learning theory*. MIT Press, 1994. URL: https://mitpress.mit.edu/books/introduction-computational-learning-theory.

49 Zander Kelley and Raghu Meka. Random Restrictions and PRGs for PTFs in Gaussian Space. In *37th Computational Complexity Conference (CCC 2022)*, Leibniz International Proceedings in Informatics (LIPIcs), pages 21:1–21:24, 2022. doi:10.4230/LIPIcs.CCC.2022.21.

50 Subhash Khot, Dor Minzer, and Muli Safra. On monotonicity testing and Boolean isoperimetric-type theorems. *SIAM Journal on Computing*, 47(6):2238–2276, 2018. doi:10.1137/16M1065872.

51 Anton J. Kleywegt, Alexander Shapiro, and Tito Homem-de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2002. doi:10.1137/S1052623499363220.

52 A. Klivans, R. O'Donnell, and R. Servedio. Learning geometric concepts via gaussian surface area. In *49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 541–550, 2008. doi:10.1109/focs.2008.64.

53 Pravesh Kothari, Amir Nayyeri, Ryan O'Donnell, and Chenggang Wu. Testing surface area. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014*. SIAM, 2014. doi:10.1137/1.9781611973402.89.

54 Abhiruk Lahiri, Ilan Newman, and Nithin Varma. Parameterized convexity testing. In *Symposium on Simplicity in Algorithms (SOSA)*, pages 174–181. SIAM, 2022. doi:10.1137/1.9781611977066.12.

55 Jane Lange, Ephraim Linder, Sofya Raskhodnikova, and Arsen Vasilyan. Local lipschitz filters for bounded-range functions with applications to arbitrary real-valued functions. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2881–2907. SIAM, 2025. doi:10.1137/1.9781611978322.93.

56 Jane Lange, Ronitt Rubinfeld, and Arsen Vasilyan. Properly learning monotone functions via local correction. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 75–86. IEEE, 2022. doi:10.1109/FOCS54457.2022.00015.

57 Jane Lange and Arsen Vasilyan. Robust learning of halfspaces under log-concave marginals. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL: https://proceedings.neurips.cc/paper_files/paper/2025/hash/91a5bb5e5939cb075f5f2464d7b8bbf0-Abstract-Conference.html.

58 Jane Lange and Arsen Vasilyan. Agnostic proper learning of monotone functions: beyond the black-box correction barrier. *SIAM Journal on Computing*, 55(3):FOCS23–1–FOCS23–32, 2026. doi:10.1137/24M1630463.

59 Grigorii Aleksandrovich Margulis. Probabilistic characteristics of graphs with large connectivity. *Problemy peredachi informatsii*, 10(2):101–108, 1974. URL: http://mi.mathnet.ru/ppi1034.

60 K. Matulef, R. O'Donnell, R. Rubinfeld, and R. Servedio. Testing halfspaces. *SIAM J. on Comput.*, 39(5):2004–2047, 2010. doi:10.1137/070707890.

61 Rahul Mazumder, Arkopal Choudhury, Garud Iyengar, and Bodhisattva Sen. A computational framework for multivariate convex regression and its variants. *Journal of the American Statistical Association*, 114(525):318–331, 2019. doi:10.1080/01621459.2017.1407771.

62 R. O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014. URL: http://www.cambridge.org/de/academic/subjects/computer-science/algorithmics-complexity-computer-algebra-and-computational-g/analysis-boolean-functions.

63 Ramesh Krishnan S Pallavoor, Sofya Raskhodnikova, and Nithin Varma. Parameterized property testing of functions. *ACM Transactions on Computation Theory (TOCT)*, 9(4):1–19, 2017. doi:10.1145/3155296.

64 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. On testing convexity and submodularity. *SIAM Journal on Computing*, 32(5):1158–1184, 2003. doi:10.1137/S0097539702414026.

65 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72(6):1012–1042, 2006. doi:10.1016/j.jcss.2006.03.002.

66 Luis Rademacher and Santosh Vempala. Testing geometric convexity. In *Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 469–480, 2005. doi:10.1007/978-3-540-30538-5_39.

67 Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951. URL: https://www.jstor.org/stable/2236626.

68 R. Tyrrell Rockafellar. *Convex Analysis*, volume 28 of *Princeton Landmarks in Mathematics and Physics*. Princeton University Press, 1970.

69 R Tyrrell Rockafellar. Lagrange multipliers and optimality. *SIAM review*, 35(2):183–238, 1993. doi:10.1137/1035044.

70 Günter Rote. The convergence rate of the sandwich algorithm for approximating convex functions. *Computing*, 48(3):337–361, 1992. doi:10.1007/BF02238642.

71 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996. doi:10.1137/S0097539793255151.

72 Emilio Seijo and Bodhisattva Sen. Nonparametric least squares estimation of a multivariate convex regression function. *The Annals of Statistics*, 39(3):1633, 2011. doi:10.1214/10-AOS852.

73 Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014. URL: http://www.cambridge.org/de/academic/subjects/computer-science/pattern-recognition-and-machine-learning/understanding-machine-learning-theory-algorithms.

74 Gyorgy Sonnevend. An optimal sequential algorithm for the uniform approximation of convex functions on $[0, 1]^2$. *Applied Mathematics and Optimization*, 10(1):127–142, 1983. doi:10.1007/BF01448382.

75 Balázs Szörényi. Characterizing statistical query learning: Simplified notions and proofs. In *Algorithmic Learning Theory, 20th International Conference, ALT*, pages 186–200. Springer, 2009. doi:10.1007/978-3-642-04414-4_18.

76 Michel Talagrand. Isoperimetry, logarithmic sobolev inequalities on the discrete cube, and margulis' graph connectivity theorem. *Geometric & Functional Analysis GAFA*, 3(3):295–314, 1993. doi:10.1007/BF01895691.

77 Michel Talagrand. How much are increasing sets positively correlated? *Combinatorica*, 16(2):243–258, 1996. doi:10.1007/BF01844850.

78 Santosh S. Vempala. Learning Convex Concepts from Gaussian Distributions with PCA. In *51th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 124–130, 2010. doi:10.1109/FOCS.2010.19.

Mixing and Cutoff for the Systematic Scan Dynamics of the Mean-Field Ferromagnetic Potts Model

Antonio Blanca
The Pennsylvania State University, University Park, PA, USA

Md Tahmidur Rafid
The Pennsylvania State University, University Park, PA, USA

Abstract

We study the mixing time of the systematic scan dynamics for the q-state ferromagnetic Potts model on the n-vertex complete graph, known as the mean-field model. This Markov chain updates vertices sequentially according to a fixed predetermined order, in contrast to the Glauber dynamics which updates a uniformly random vertex at each step. Systematic scan dynamics are attractive in practice as they often demonstrate strong empirical performance. However, their theoretical analysis remains far less developed than that of the Glauber dynamics.

We take a step toward addressing this imbalance by showing that for every $q \geq 2$ and $\beta < \beta_s$, where β_s is the metastability threshold associated with the onset of slow mixing for the Glauber dynamics, the systematic scan dynamics for the ferromagnetic mean-field Potts model mixes in $\Theta(\log n)$ scans or, equivalently, in $\Theta(n \log n)$ single site updates. We in fact prove a sharper result; namely, that there exists a constant $c(\beta, q) > 0$ such that the mixing time is $c(\beta, q) \log n + \Theta(1)$, which implies that the Markov chain exhibits the cutoff phenomenon, with the total variation distance to the stationary distribution dropping abruptly from nearly 1 to nearly 0 within a narrow $\Theta(1)$ time window. This result is tight in β as well since the dynamics mixes exponentially slowly for $\beta > \beta_s$. To the best of our knowledge, this is the first general cutoff result for the systematic scan dynamics in the context of spin systems. The result may also be of independent interest in the theory of Markov chains, since the systematic scan dynamics is both global and non-reversible, two settings in which cutoff remains poorly understood.

2012 ACM Subject Classification Theory of computation → Random walks and Markov chains

Keywords and phrases Markov chain, mixing times, systematic scan, Potts model

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.63

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2607.09841 [5]

Funding Research supported in part by NSF CAREER grant CCF-2143762.

1 Introduction

Sampling high-dimensional probability distributions is an important task in science and engineering. It is, for example, a prevalent problem when running simulations in statistical physics or when solving inference problems in machine learning. As such, there is an array of sophisticated algorithms and heuristics to generate samples from these distributions. One of the most classical and powerful approaches is *Gibbs Sampling*, which iteratively updates one variable at a time conditioned on the values of all other variables.

There are two standard strategies for deciding which variable to update in each step: (i) the Glauber dynamics, where the variable to be updated is chosen uniformly at random, and (ii) systematic scan dynamics, where variables are updated according to a predetermined ordering. In practice, systematic scan dynamics offer several computational advantages over


© Antonio Blanca and Md Tahmidur Rafid;
licensed under Creative Commons License CC-BY 4.0

the Glauber dynamics, as they are often amenable to parallelization, can exhibit improved memory locality and thus reduce data-access overhead, and require fewer random bits [29]. Despite these practical advantages, most theoretical analyses focus on the Glauber dynamics, which have historically proven more mathematically tractable, leaving the convergence theory for systematic scan dynamics significantly less developed.

The number of variable updates required to reach stationarity can differ significantly between the Glauber and systematic scan dynamics, sometimes up to polynomial factors in n [21, 27]. For example, even in the trivial case of n independent variables, the systematic scan mixes in exactly n updates, whereas the Glauber dynamics requires $\frac{1}{2}n\log n$ updates. When combined with the implementation benefits noted above, such differences in convergence rates can make the systematic scan dynamics a particularly attractive update strategy.

In this paper, we take a step toward a better theoretical understanding of the systematic scan dynamics by studying them in the context of the *ferromagnetic mean-field Potts model*, a classical spin system model central in statistical physics, theoretical computer science, machine learning, and probability theory. In this model, each variable in $V = \{1, \dots, n\}$ can take a value from $Q = \{1, ..., q\}$. The Gibbs or Boltzmann distribution assigns to each configuration $\sigma \in \Sigma_n = Q^V$ a probability given by:

$$\mu(\sigma) = \frac{1}{Z_{\beta,n}} \exp\left\{\frac{\beta}{n} \sum_{\{u,v\}\subseteq V} \mathbf{1}\big(\sigma(u) = \sigma(v)\big)\right\},$$

where $\sigma(u)$ is the value (or spin) assigned to vertex u in the configuration σ, and $Z_{\beta,n}$ is the normalization constant or partition function. The parameter β captures the strength of the interaction between the variables and is proportional to the inverse temperature in statistical physics applications. The $q = 2$ case corresponds to the classical Ising model. The Potts model can be defined on any graph by taking the summation in the definition over the edges of the graph instead of over pairs of vertices; the mean-field setting considered here corresponds to the complete graph on n vertices.

For concreteness, we consider the standard variant of the systematic scan dynamics in which site updates are given by the *heat-bath* update, i.e., they are performed according to the conditional distribution at each update site induced by the configuration on the remaining vertices. Formally, if vertex v is to be updated in a configuration σ, then v is assigned a new spin sampled from $\mu(\cdot \mid \sigma(V \setminus \{v\}))$. This is a non-reversible ergodic Markov chain with state space Σ_n that converges to μ. We measure rates of convergence to μ in terms of the *mixing time*, which is the most standard notion of speed of convergence for a Markov chain. It is defined as the number of steps until the dynamics is close in total variation distance to μ starting from any initial configuration. Formally, if P denotes the transition matrix of the chain, for $\varepsilon \in (0, 1)$, the ε-mixing time is given by

$$T_{\mathrm{mix}}(\varepsilon) = \min\left\{t \in \mathbb{Z}_{\geq 0} : \max_{\sigma\in\Sigma_n} \left\|P^t(\sigma, \cdot) - \mu\right\|_{\mathrm{TV}} \leq \varepsilon\right\}.$$

By convention, the mixing time is taken to be $T_{\mathrm{mix}} = T_{\mathrm{mix}}(1/4)$.

A more refined notion of convergence, when present, is the cutoff phenomenon. A Markov chain exhibits *cutoff* when there is a sharp drop in the total variation distance from close to 1 to close to 0 in an interval of time of smaller order than its mixing time; which is called the *cutoff window*. More precisely, we say that there is cutoff if as $n \to \infty$, for any fixed $\varepsilon \in (0, 1)$

$$T_{\mathrm{mix}}(\varepsilon) - T_{\mathrm{mix}}(1 - \varepsilon) = o(T_{\mathrm{mix}}). \tag{1}$$

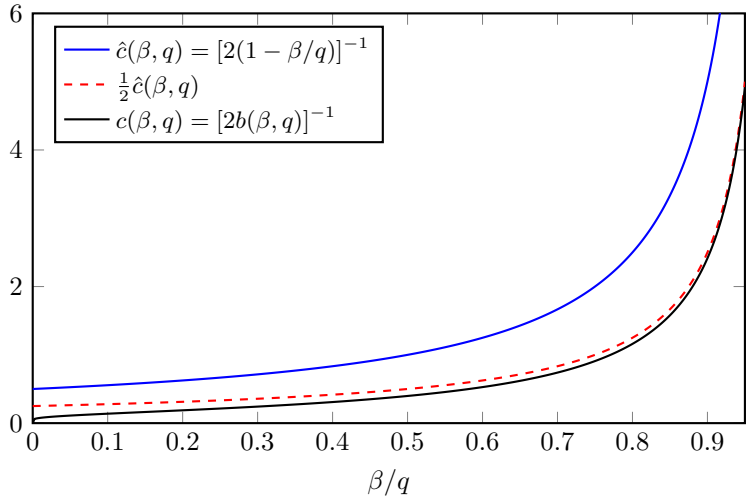


Figure 1 Comparison of mixing times of Glauber (blue curve) and systematic scan dynamics (black curve).

The mixing time and cutoff of the Glauber dynamics for the mean-field ferromagnetic Potts model is by now very well-understood. In the case of the Ising model (the $q = 2$ case) at $\beta < 2$, the Glauber dynamics mixes in $\frac{1}{2(1-\beta/2)} n \log n$ steps with an $O(n)$ cutoff window. The mixing time is $\Theta(n^{3/2})$ at $\beta = 2$ and exponential in n when $\beta > 2$ [13, 12, 23]. For $q \geq 3$, the model exhibits a first-order phase transition and the dynamics undergoes a slowdown at $\beta_s = \beta_s(q)$, which is the spinodal point marking the onset of metastability. Specifically, when $\beta < \beta_s$ the mixing time is $\frac{1}{2(1-\beta/q)} n \log n$ with an $O(n)$ cutoff window. At $\beta = \beta_s$ the dynamics mixes in $\Theta(n^{4/3})$ and no longer exhibits cutoff, and the mixing time is again exponential in n for $\beta > \beta_s$ [10].

In contrast with this fairly complete convergence picture for the mean-field Glauber dynamics, much less is known for systematic scan dynamics, with the best known mixing time bound being $O(n^2 \log n)$ for $\beta < \beta_s$. We provide here the first sharp analysis of the mixing time of the systematic scan dynamics for the mean-field ferromagnetic Potts model, valid for all scan orderings and for all inverse temperatures $\beta < \beta_s$.

▶ **Theorem 1.** *For $q \geq 2$, $\beta < \beta_s$, there exists a constant $c(\beta, q) > 0$ such that the systematic scan dynamics for the mean-field ferromagnetic Potts model exhibits cutoff at mixing time $T_{mix} = c(\beta, q) \log(n) + \Theta(1)$.*

For n large, the leading constant $c(\beta, q)$ corresponds to $\frac{1}{2b(\beta,q)}$ where $b = b(\beta, q)$ is the root of the equation $\beta(1 - e^{-b})/(qb) = e^{-b}$; see Section 4 for additional details. Figure 1 shows a plot of the constant $c(\beta, q)$ as a function of β/q and compares it to the leading constant $\hat{c}(\beta, q) = \frac{1}{2(1-\beta/q)}$ for the mixing time of the Glauber dynamics, revealing at least a twofold speedup. This is consistent with the folklore conjecture that the systematic scan dynamics exhibits such a speedup in several settings.

We note that Theorem 1 is tight in β, since, as mentioned, the Glauber dynamics mixes exponentially slowly for $\beta > \beta_s$ and known mixing time comparison results (see, e.g., [17, Theorem 1.2]) imply that so does the systematic scan. At $\beta = \beta_s$, based on the known behavior of the Glauber dynamics, we conjecture that the systematic scan dynamics mixes $\Theta(n^c)$ for some $c < 1$ without cutoff.

To better contextualize Theorem 1, we briefly comment on prior work on mixing times for the systematic scan dynamics on spin systems. There is a line of work that seeks to compare the convergence rates of Glauber and systematic scan dynamics. This is done with varying degrees of generality in [2, 17, 14, 11, 8, 9, 19]. In parallel, there is work establishing fast mixing for the systematic scan dynamics under model parameter and graph-structural conditions [15, 20, 16], or under monotonicity assumptions [7, 3, 26]. We note that the recent advances in Markov chain mixing via spectral independence have not yet yielded major new results for this dynamics; see [1] for initial progress. This body of work demonstrates sustained interest on the systematic scan dynamics while also underscoring the significant analytical challenges they present, with many questions that are well understood for the Glauber dynamics remaining unresolved in this setting.

Turning to cutoff, results like Theorem 1 are rarer still. To the best of our knowledge, our work provides the first cutoff result for the systematic scan dynamics in the context of spin systems. The cutoff result in [22] for the Ising model considers a Markov chain inspired by systematic scan but that is effectively quite different since in each step, $k = o(n^{1/3})$ vertices are chosen uniformly random and updated according to a random permutation. More generally, the systematic scan dynamics is a global, non-reversible Markov chain, and both features introduce substantial additional analytical challenges. Indeed, the only other cutoff result for a global Markov chain for spin systems we are aware of is the recent work [6] on the Swendsen-Wang dynamics, while the study of the cutoff phenomenon for non-reversible Markov chains is only in its early stages [25].

Proof Sketch

For the rest of the paper, we assume $\beta < \beta_s$, $q \geq 2$ and n to be asymptotically large. We refer to the vertex updates of the dynamics as *site updates*, and n consecutive vertex updates are called a *full-scan*. A step of the systematic scan dynamics Markov chain corresponds to a full-scan starting from the first vertex in the ordering. It will be convenient to index the site updates, and thus we use σ_t for the configuration after the t-th site update and σ_{kn} for the configuration after the k-th step of the chain.

We will show that for any fixed $\varepsilon \in (0,1)$, there exists a $\kappa(\varepsilon)$ independent of n such that

$$c(\beta,q)\log n - \kappa(\varepsilon) \leq T_{\mathrm{mix}}(\varepsilon) \leq c(\beta,q)\log n + \kappa(\varepsilon). \tag{2}$$

From this, Theorem 1 follows immediately; in particular, this inequality implies (1).

To establish the upper bound on $T_{\mathrm{mix}}(\varepsilon)$ we utilize a multiphase *coupling*, and in that sense, it is reminiscent of the other Markov chain analysis in the mean-field setting [18, 4, 6, 10]. A coupling considers two copies, $\{\sigma_t\}_{t\geq 0}$ and $\{\tilde{\sigma}_t\}_{t\geq 0}$, of the systematic scan dynamics starting from two arbitrary initial configurations σ_0 and $\tilde{\sigma}_0$, respectively. Each copy evolves according to the transition matrix P, but the moves of the two copies can be arbitrarily correlated. The ε-coupling time $T_{\mathrm{coup}}(\varepsilon)$ is the minimum number of full-scans T such that $\Pr[\sigma_{Tn} \neq \tilde{\sigma}_{Tn}] \leq \varepsilon$. A standard result says that $T_{\mathrm{mix}}(\varepsilon) \leq T_{\mathrm{coup}}(\varepsilon)$; for additional details see [24].

To describe the three main phases of our coupling we introduce some useful notation first. For a configuration $\sigma \in \Sigma_n$, let $\alpha(\sigma)$ denote the q-dimensional vector with coordinates

$$\alpha_i(\sigma) = \frac{1}{n}\sum_{j=1}^{n} \mathbf{1}(\sigma(j) = i).$$

We call $\alpha(\sigma)$ the proportions vector of σ. We let $\hat{e} = \big(1/q, \ldots, 1/q\big)$ denote the q-dimensional equiproportions vector that corresponds to the configuration with exactly the same number of vertices assigned each spin (ignoring rounding issues).

The main phases of our coupling are the following:

Phase 1: From arbitrary initial configurations σ_0 and $\tilde{\sigma}_0$, run two copies $\{\sigma_t\}$ and $\{\tilde{\sigma}_t\}$ independently for an initial *burn-in* period of $T = O(1)$ scans. After this initial burn-in, we show that both copies are such that $\|\alpha(\sigma_{Tn}) - \hat{e}\|_\infty \leq \rho$ and $\|\alpha(\tilde{\sigma}_{Tn}) - \hat{e}\|_\infty \leq \rho$ with probability $1 - o(1)$, for any desired constant $\rho > 0$. In addition, we show that both copies have "good spread" of their spins with respect to the ordering of the systematic scan dynamics.

Phase 2: We can then couple the evolution of the two instances of Markov chain so that after $c(\beta, q) \log n + O(1)$ additional scans: the Hamming distance between the two configurations is $O(\sqrt{n})$ and the ℓ_2 distance between their respective proportions vector and $\hat{e}$ is also $O(1/\sqrt{n})$.

Phase 3: Finally, we couple one final scan in a way that guarantees that the processes coalesce with at least the desired ε probability.

We provide additional details about **Phase 1** next. Let $e_i \in \mathbb{R}^q_{\geq 0}$ be the indicator vector with value 1 in the i-th coordinate and 0 elsewhere. We capture the notion of closeness to $\hat{e}$ with "good spread" by considering for each $\rho > 0$ the set of configurations

$$\Sigma_n^\rho = \left\{\sigma \in \Sigma_n : \left\|\hat{e} - \frac{1}{i}\sum_{j=n-i+1}^{n} e_{\sigma(j)}\right\|_\infty \leq \rho \quad \forall\, i \in \mathbb{N} \cap (n^{3/4}, n]\right\}.$$

Observe that $\sigma_t \in \Sigma_n^\rho$ implies not only that $\|\alpha(\sigma_t) - \hat{e}\|_\infty \leq \rho$, but also that as we perform site updates to generate σ_{t+n}, the intermediate configurations are also balanced. To see this, note that when updating σ_j for $j \in [t, t+n]$, the fact that $\sigma_t \in \Sigma_n^\rho$ ensures that the partial configuration in the vertices $v_{j+1-t}, \ldots, v_n$ is also balanced; in addition, the partial configuration in $v_1, \ldots, v_{j-t-1}$ we can guarantee to be balanced since it was just updated. (The $n^{3/4}$ gives a required slack to make such statement possible; note, for example, that a single vertex cannot be balanced.)

This more fine grained notion of closeness to $\hat{e}$ marks a clear distinction between the Glauber and systematic scan analyses. In the former, given a proportions vector α, the dynamics is oblivious to the specific spin assignment to vertices (i.e., from $\alpha(\sigma_{t-1})$ we can generate $\alpha(\sigma_t)$); as such, the speed of convergence of the Markov chain is governed by its ℓ_∞ closeness to $\hat{e}$. However, this is not the case for systematic scan, since this Markov chain updates vertices in a fixed order and thus depends on the specific assignment of spins to vertices along the scan order (not only on the spin counts). We are therefore required to establish this stronger notion of closeness to $\hat{e}$. We show that the following holds after **Phase 1**.

▶ **Lemma 2.** *For any fixed $\rho > 0$, there exist constants $c > 0$ and $k \in \mathbb{Z}_+$ depending only on β, q, and ρ such that*

$$\Pr\left[\sigma_{kn} \notin \Sigma_n^\rho\right] \leq 2qke^{-c\sqrt{n}}. \tag{3}$$

Moreover, if $\sigma_0 \in \Sigma_n^\rho$, for any $\hat{k} \in \mathbb{Z}_+$ we have

$$\Pr\left[\max_{1 \leq i \leq n} \left\|\alpha(\sigma_{(\hat{k}-1)n+i}) - \hat{e}\right\|_\infty > q\rho\right] \leq 2q\hat{k}e^{-c\sqrt{n}}. \tag{4}$$

We note that (3) establishes that the chain enters the set Σ_n^ρ after a constant number of scans with high probability, whereas (4) shows that once the chain enters Σ_n^ρ, it will remain also with high probability. To prove (3), we show that the proportions vector has drift

towards the $\hat{e}$. More precisely, we analyze the evolution of fraction of vertices in a spin class by decomposing into a drift and a fluctuation component. The drift component of each coordinate is then bounded by a one-dimensional recursive function that iterates towards its fixed point $1/q$. For every fixed constant $\rho > 0$ (independent of n), the recursive function enters the interval $[1/q, 1/q + \rho]$ after a constant number of iterations. To analyze the fluctuation component of the process, we show first that it is a supermartingale, and then we use the Doob decomposition to split it into a predictable non-increasing process and a martingale, with the martingale fluctuations controlled via a concentration of its maximum over an interval; more precisely, via a Maximal-Azuma inequality, a tailored version of which we establish here.

The proof of (4) follows from a similar argument. Once the proportions vector is close to $\hat{e}$ and the configuration has a good spread, the drift helps the process to remain close to $\hat{e}$. Then, the corresponding martingale estimate gives the desired high probability statement. The proof of Lemma 2 is fleshed out in Section 2.

In **Phase 2**, we couple the systematic scan dynamics using the optimal single site coupling; i.e., the coupling that maximizes the probability of agreement between the two copies of the chain at the update site conditional on the configuration off the site. We are able to establish that the Hamming distance d_H between the two chains drops from $O(n)$ to $O(\sqrt{n})$ in $O(\log n)$ steps, and that at the same time, the distance of the spin fraction from $\hat{e}$ drops from $O(1)$ to $O(1/\sqrt{n})$.

▶ **Lemma 3.** *Let $\rho > 0$ small enough and $\sigma_0, \tilde{\sigma}_0 \in \Sigma_n^\rho$. Then, for any fixed constants $C, \varepsilon > 0$, there exist an integer $k = k(\beta, q, C, \varepsilon)$ such that with probability at least ε for $T = c(\beta, q)\log(n) + k$ we have $d_H(\sigma_{Tn}, \tilde{\sigma}_{Tn}) \leq C\sqrt{n}$.*

To establish this lemma we show that for two copies of the systematic scan Markov chain with starting configurations in Σ_n^ρ, their Hamming distance exhibits an almost uniform geometric decay. To establish this, we show that the distances between the proportions vectors of the copies and $\hat{e}$ contract at exactly same geometric rate. Identifying these matching contraction rates determines the number of steps required for the Hamming distance to reach the desired scale (i.e., $O(\sqrt{n})$) and determines the leading constant $c(\beta, q)$ in the mixing time. The full proof of Lemma 3 is given in Sections 3, 4, and 5.

For **Phase 3** of the coupling we show that the relative entropy between the distribution of σ_{t+n} and $\tilde{\sigma}_{t+n}$ can be bounded in terms of the square of the Hamming distance between σ_t and $\tilde{\sigma}_t$. Then, since we start the phase from two configurations within $O(\sqrt{n})$ Hamming distance, Pinsker's inequality allows to conclude the following.

▶ **Lemma 4.** *There exists a coupling such that for any $\varepsilon > 0$ there is a constant $C = C(\varepsilon, \beta, q)$ such that for $\sigma_0, \tilde{\sigma}_0 \in \Sigma_n$ with $d_H(\sigma_0, \tilde{\sigma}_0) \leq C\sqrt{n}$, we have $\Pr\left[\sigma_n \neq \tilde{\sigma}_n\right] \leq \varepsilon$.*

The proof of Lemma 4 is deferred to the full version of the paper [5]. We can now provide the proof of the mixing time upper bound in Theorem 1.

Proof of Theorem 1 (upper bound). We show that $T_{\text{mix}}(\varepsilon) \leq c(\beta, q)\log n + \kappa(\varepsilon)$ for any $\varepsilon \in (0, 1)$ and a suitable κ independent of n. By Lemma 2, for any $\rho > 0$ there exists $\kappa_1 = \kappa_1(\beta, q, \rho)$ such that $\sigma_{\kappa_1 n}, \tilde{\sigma}_{\kappa_1 n} \in \Sigma_n^\rho$ with probability at least $1 - \varepsilon/3$. Conditioning on $\sigma_{\kappa_1 n}, \tilde{\sigma}_{\kappa_1 n} \in \Sigma_n^\rho$ and choosing ρ small enough, Lemma 3 implies that we can couple the the two copies of the chain so that for any fixed $C > 0$ there exists $\kappa_2 = \kappa_2(\beta, q, C, \varepsilon)$ such that after $T_2 = c(\beta, q)n\log n + \kappa_2 n + \kappa_1 n$ site updates we have that

$$\Pr\left[d_H(\sigma_{T_2}, \tilde{\sigma}_{T_2}) \leq C\sqrt{n}\right] \geq 1 - \varepsilon/3.$$

For the last phase, Lemma 4 implies that $\sigma_{T_2+n} = \tilde{\sigma}_{T_2+n}$ with probability at least $1 - \varepsilon/3$ for small enough C. So at $T = T_2 + n$ we obtain $\Pr[\sigma_T \neq \tilde{\sigma}_T] \leq \frac{\varepsilon}{3} + \frac{\varepsilon}{3} + \frac{\varepsilon}{3} = \varepsilon$. ◀

To establish the mixing time lower-bound in (2), we use the fact that the distribution μ is well-concentrated around the equiproportion vector $\hat{e}$. Therefore, if the event $\{\|\alpha(\sigma_{kn})-\hat{e}\|_2 > C/\sqrt{n}\}$ has a significant probability for a sufficiently large constant $C > 0$, the systematic scan dynamics chain cannot be mixed after k steps. Using our precise understanding of the rate of contraction towards $\hat{e}$ established in the analysis of **Phase 2**, and standard concentration inequalities, we prove that for any $C > 0$ and any $\varepsilon \in (0,1)$, there exists κ independent of n such that after $c(\beta,q)\log n - \kappa(\varepsilon, C)$ steps $\Pr[\|\alpha(\sigma_t)-\hat{e}\|_2 > C/\sqrt{n}] > 1-\varepsilon$ where for any fixed ε the function $\kappa(\varepsilon,\cdot)$ is increasing. The lower bound proof is provided in the full version of the paper [5].

2 Independent Burn-in

Our aim in this section is to analyze the **Phase 1** of our coupling and establish Lemma 2. We analyze the evolution of the number of vertices with a given spin (say spin 1 for concreteness) by decomposing this one-dimensional stochastic process into a mostly predictable component determined by a drift function and a fluctuation component that is controlled via martingale concentration bounds.

2.1 Drift analysis

Consider the drift function $G_\beta : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ given by

$$G_\beta(x) = \frac{e^{\beta x}}{e^{\beta x} + (q-1)e^{\beta\frac{1-x}{q-1}}}.$$

This function approximates the probability that a single-site update chooses spin 1 in a configuration that has xn vertices assigned spin 1 and all other $q-1$ spins distributed equally among the remaining $(1-x)n$ vertices. The correspondence is not exact in part because the spin of the selected vertex is taken into consideration; to account for this correction we define

$$G_{\beta,n}(x) = G_\beta(x) + \beta/n.$$

For $q \geq 2$, the spinodal point β_s can be defined in terms of the drift function G_β:

$$\beta_s = \sup\left\{\beta > 0 : G_\beta(x) - x \neq 0 \quad \forall x \in (1/q, 1)\right\};$$

see [10] for additional details.

We use the function G_β to control the evolution of the one-dimensional process corresponding to fraction of vertices assigned spin 1. To formalize this, let J_t be the random variable corresponding to the spin chosen in the t-th site update and let $X_t = \mathbf{1}(J_t = 1)$. We use the convention that for $-n < t \leq 0$, X_t indicates whether the $(t+n)$-th vertex in the initial configuration σ_0 has spin 1. We define $Y_t^{(\ell)}$ as the average of the X_i's for the last ℓ updates:

$$Y_t^{(\ell)} = \frac{1}{\ell}\sum_{i=t-\ell+1}^{t} X_i. \tag{5}$$

Observe that $Y_t^{(n)} = \alpha_1(\sigma_t)$; to simplify the notation we set $Y_t = Y_t^{(n)}$. We decompose the process $Y_t^{(\ell)}$ into the sum of a drift and a fluctuation component as follows. For all $t \geq \ell$ and $1 \leq \ell \leq n$:

$$Y_t^{(\ell)} = \frac{1}{\ell}\sum_{j=t-\ell+1}^{t} G_{\beta,n}(Y_{j-1}) + \frac{1}{\ell}\sum_{j=t-\ell+1}^{t}\left(X_j - G_{\beta,n}(Y_{j-1})\right),$$

and rewrite the fluctuation term as

$$D_t^{(\ell)} = \sum_{j=t-\ell+1}^{t} \big(X_j - G_{\beta,n}(Y_{j-1})\big), \tag{6}$$

so that

$$Y_t^{(\ell)} = \frac{1}{\ell} D_t^{(\ell)} + \frac{1}{\ell} \sum_{j=t-\ell+1}^{t} G_{\beta,n}(Y_{j-1}).$$

Our first lemma shows that under the assumption that the random fluctuations are small in aggregate over all intervals between kn and $(k+1)n$, then $\max_{n^{3/4}<i\le n} Y_{kn}^{(i)}$ has a contraction given by the function $G_{\beta,n}$ up to a small correction.

▶ **Lemma 5.** *Suppose that for some integer $k \ge 0$, a fixed constant $a \in (1/q, 1]$, and a sufficiently small $\varepsilon > 0$ that satisfies $G_{\beta,n}(a) + 2\varepsilon < a$, we have $\max_{n^{3/4}<i\le n} Y_{kn}^{(i)} \le a$, and $\max_{1\le i\le j\le n} D_{kn+j}^{(i)} < \varepsilon n^{3/4}$. Then,*

$$\max_{n^{3/4}<i\le n} Y_{(k+1)n}^{(i)} \le 2\varepsilon + G_{\beta,n}(a).$$

Moreover, $\max_{1\le i\le n} Y_{kn+i} \le a + \frac{1}{n^{1/4}}$.

Proof. We start by rewriting $Y_{kn+\ell}$ using (5) and (6) as

$$\begin{aligned}
Y_{kn+\ell} &= \frac{1}{n} \sum_{i=(kn+\ell)-n+1}^{kn} X_i + \frac{1}{n} \sum_{i=kn+1}^{kn+\ell} X_i \\
&= \frac{1}{n} \sum_{i=kn-(n-\ell)+1}^{kn} X_i + \frac{1}{n} \sum_{i=kn+1}^{kn+\ell} \Big(X_i - G_{\beta,n}(Y_{i-1})\Big) + \frac{1}{n} \sum_{i=kn+1}^{kn+\ell} G_{\beta,n}(Y_{i-1}) \\
&= \frac{(n-\ell)}{n} Y_{kn}^{(n-\ell)} + \frac{1}{n} D_{kn+\ell}^{(\ell)} + \frac{\ell}{n} \frac{1}{\ell} \sum_{i=kn+1}^{kn+\ell} G_{\beta,n}(Y_{i-1}) \\
&= \frac{1}{n} D_{kn+\ell}^{(\ell)} + \frac{1}{n} \bigg((n-\ell) Y_{kn}^{(n-\ell)} + \ell \Big(\frac{1}{\ell} \sum_{i=kn+1}^{kn+\ell} G_{\beta,n}(Y_{i-1}) \Big) \bigg).
\end{aligned} \tag{7}$$

We show first via induction on ℓ that

$$Y_{kn+\ell} \le a + \frac{1}{n^{1/4}}. \tag{8}$$

for all $\ell = 1, \dots, n$. This ensures that $Y_{kn+\ell}$ does not deviate much from its initial upper bound of a on the process. Conditioned on this, we can establish the desired contraction $\max_{n^{3/4}<i\le n} Y_{(k+1)n}^{(i)} \le 2\varepsilon + G_{\beta,n}(a)$.

For $\ell = 1$, under the assumptions of the lemma, and using the fact that $G_{\beta,n}$ is a monotonically increasing function, we obtain from (7)

$$Y_{kn+1} < \frac{\varepsilon n^{3/4}}{n} + \frac{1}{n}\big((n-1)a + G_{\beta,n}(a)\big) \le \frac{\varepsilon}{n^{1/4}} + a.$$

For the induction step we consider two cases. First assume that (8) holds for all $\ell \in \{1, \dots, \ell_1 - 1\}$ for some $\ell_1 \in \{1, \dots, n - n^{3/4} - 1\}$. From (7), the monotonicity of $G_{\beta,n}$, the lemma assumptions, and the inductive hypothesis, we get

$$\begin{aligned} Y_{kn+\ell_1} &= \frac{1}{n} D^{(\ell_1)}_{kn+\ell_1} + \frac{1}{n}\Big((n-\ell_1)Y^{(n-\ell_1)}_{kn} + \ell_1\Big(\frac{1}{\ell_1}\sum_{i=kn+1}^{kn+\ell_1} G_{\beta,n}(Y_{i-1})\Big)\Big) \\ &< \frac{\varepsilon}{n^{1/4}} + \frac{1}{n}\Big((n-\ell_1)a + \ell_1 G_{\beta,n}\Big(a + \frac{1}{n^{1/4}}\Big)\Big). \end{aligned}$$

Now, by the mean value theorem, since the derivative of $G_{\beta,n}$ is bounded in the interval $\big[a, a + \frac{1}{n^{1/4}}\big]$, we have $G_{\beta,n}(a + \frac{1}{n^{1/4}}) = G_{\beta,n}(a) + O(\frac{1}{n^{1/4}}) \le a - \varepsilon$ for large enough n. Then,

$$\begin{aligned} Y_{kn+\ell_1} &\le \frac{\varepsilon}{n^{1/4}} + \frac{1}{n}\Big((n-\ell_1)a + \ell_1(a-\varepsilon)\Big) \\ &\le \frac{\varepsilon}{n^{1/4}} + \frac{1}{n}\big(na - \ell_1\varepsilon\big) = a + \frac{\varepsilon}{n^{1/4}} - \frac{\ell_1}{n}\varepsilon \le a + \frac{1}{n^{1/4}}. \end{aligned}$$

Thus, the claim holds for all $1 \le \ell < n - n^{3/4}$. For the remaining range, let us assume that the claim holds for all $\ell \in \{1, \dots, \ell_2 - 1\}$ for some $\ell_2 \in \{n - n^{3/4} - 1, \dots, n\}$. Then, similarly

$$\begin{aligned} Y_{kn+\ell_2} &= \frac{1}{n} D^{(\ell_2)}_{kn+\ell_2} + \frac{1}{n}\Big((n-\ell_2)Y^{(n-\ell_2)}_{kn} + \ell_2\Big(\frac{1}{\ell_2}\sum_{i=kn+1}^{kn+\ell_2} G_{\beta,n}(Y_{i-1})\Big)\Big) \\ &< \frac{\varepsilon}{n^{1/4}} + \frac{1}{n}\Big(n - \ell_2 + \ell_2 G_{\beta,n}\Big(a + \frac{1}{n^{1/4}}\Big)\Big) \\ &\le \frac{\varepsilon}{n^{1/4}} + \frac{1}{n}\Big((n^{3/4}+1) + n(a-\varepsilon)\Big) = a - \varepsilon + \frac{\varepsilon}{n^{1/4}} + \frac{1}{n^{1/4}} + \frac{1}{n} \le a, \end{aligned}$$

and thus (8) follows.

To establish that $\max_{n^{3/4} < i \le n} Y^{(i)}_{(k+1)n} \le 2\varepsilon + G_{\beta,n}(a)$, observe that the lemma assumptions and (8) imply for all $\ell \in \{n^{3/4} + 1, \dots, n\}$,

$$\begin{aligned} Y^{(\ell)}_{(k+1)n} &= \frac{1}{\ell} D^{(\ell)}_{(k+1)n} + \frac{1}{\ell}\sum_{j=(k+1)n-\ell+1}^{(k+1)n} G_{\beta,n}(Y_{j-1}) \\ &< \frac{\varepsilon n^{3/4}}{\ell} + \frac{1}{\ell}\sum_{j=(k+1)n-\ell+1}^{(k+1)n} G_{\beta,n}\Big(a + \frac{1}{n^{1/4}}\Big) \le 2\varepsilon + G_{\beta,n}(a), \end{aligned}$$

which proves our claim. ◀

2.2 Fluctuation analysis

Our goal now is to bound the fluctuations $D^{(\ell)}_t$ of the $Y^{(\ell)}_t$ process. For this, consider the process

$$M_t = \sum_{i=1}^{t} (X_i - G_{\beta,n}(Y_{i-1}));$$

observe that $D^{(\ell)}_t = M_t - M_{t-\ell}$. We establish first that the process M_t is a supermartingale.

▶ **Lemma 6.** *$\{M_t\}_{t>0}$ is a supermartingale.*

The proof of this lemma is provided in Section 2.4. We bound the maximum supermartingale difference $M_i - M_j$ over a time interval using a Maximal-Azuma inequality. We use a small variation of a version of this type inequality that appears in [28]. A proof is provided in the full version of the paper [5] for completeness.

▶ **Lemma 7.** *Let $\{Z_t\}_{t\geq 1}$ be a bounded discrete-time supermartingale. Suppose there exists $R \in \mathbb{R}_+$ such that $|Z_{t+1} - Z_t| \leq R$ for all integer $t \geq 1$. Then, for any positive integers t_1, t_2 such that $1 \leq t_1 < t_2$ and $d > 0$ we have*

$$\Pr\left[\max_{t_1 \leq i < j \leq t_2} Z_j - Z_i > d\right] \leq 2e^{-\frac{d^2}{32(t_2-t_1)R^2}}.$$

2.3 Combining drift and fluctuation analysis

The next ingredient of the proof is to argue that repeated applications of Lemma 5 decrease the distance to equiproportionality. For this, we introduce the function $G^{\varepsilon}_{\beta,n} : \mathbb{Z}_+ \to \mathbb{R}_+$ defined recursively by setting $G^{\varepsilon}_{\beta,n}(1) = 1 - \varepsilon$, and for all integer $k \geq 2$:

$$G^{\varepsilon}_{\beta,n}(k) = G_{\beta,n}\left(G^{\varepsilon}_{\beta,n}(k-1) + \varepsilon\right). \tag{9}$$

The function $G^{\varepsilon}_{\beta,n}(k)$ is monotonically decreasing with k and for any $\rho > 0$, it enters the interval $[1/q, 1/q + \rho]$ after constant number of recursive steps. The latter is established in the following lemma which is proved in the full version of the paper [5].

▶ **Lemma 8.** *For any fixed $\rho > 0$, there exist constants $k \in \mathbb{Z}_+$ and $\varepsilon > 0$ such that $G^{\varepsilon}_{\beta,n}(k) + \varepsilon < \frac{1}{q} + \rho$.*

We use the recursive function $G^{\varepsilon}_{\beta,n}$ to probabilistically bound the process $Y^{(\ell)}_{kn}$.

▶ **Lemma 9.** *For any integer $k > 0$ and sufficiently small $\varepsilon > 0$, we have*

$$\Pr\left[\max_{n^{3/4} < i \leq n} Y^{(i)}_{kn} > G^{\varepsilon}_{\beta,n}(k) + \varepsilon\right] \leq 2ke^{-\frac{\varepsilon^2\sqrt{n}}{128}}.$$

Proof. Consider the events

$$A_k = \left\{\max_{n^{3/4} < i \leq n} Y^{(i)}_{kn} \leq G^{\varepsilon}_{\beta,n}(k) + \varepsilon\right\}, \text{ and}$$
$$B_k = \left\{\max_{1 \leq i \leq j \leq n} D^{(i)}_{(k-1)n+j} \leq \frac{\varepsilon n^{3/4}}{2}\right\}.$$

For $k > 1$ we have

$$\Pr[\overline{A}_k] \leq \Pr\left[\overline{A}_k \mid A_{k-1}\right] + \Pr[\overline{A}_{k-1}] \leq \Pr[\overline{A}_1] + \sum_{\ell=2}^{k} \Pr\left[\overline{A}_\ell \mid A_{\ell-1}\right]. \tag{10}$$

Since

$$\Pr[\overline{A}_1] = \Pr\left[\max_{n^{3/4} < i \leq n} Y^{(i)}_{n} > G^{\varepsilon}_{\beta,n}(1) + \varepsilon\right] = \Pr\left[\max_{n^{3/4} < i \leq n} Y^{(i)}_{n} > 1\right] = 0,$$

it suffices to prove that

$$\Pr[\overline{A_\ell} \mid A_{\ell-1}] \leq 2e^{-\frac{\varepsilon^2\sqrt{n}}{128}},$$

for all $\ell \in \{2, \dots, k\}$. For any such ℓ, using the definition of $G^{\varepsilon}_{\beta,n}$ in (9) and taking $a = G^{\varepsilon}_{\beta,n}(\ell-1)+\varepsilon$, we obtain from Lemma 5 that $A_{\ell-1} \cap B_\ell \subseteq A_\ell$ and thus $\overline{A}_\ell \subseteq \overline{A}_{\ell-1} \cup \overline{B}_\ell$. Then,

$$\begin{aligned}\Pr\left[\overline{A}_\ell \mid A_{\ell-1}\right] &\le \Pr\left[\overline{A}_{\ell-1} \cup \overline{B}_\ell \mid A_{\ell-1}\right] \le \Pr\left[\overline{B}_\ell \mid A_{\ell-1}\right] \\ &= \Pr\left[\max_{1\le i\le j\le n} M_{(\ell-1)n+j} - M_{(\ell-1)n+j-i} > \frac{\varepsilon n^{3/4}}{2} \,\middle|\, A_{\ell-1}\right]. \end{aligned} \tag{11}$$

We consider the following subset of configurations,

$$\Sigma^a_{n,1} = \Big\{\sigma \in \Sigma_n : \frac{1}{i}\sum_{j=n-i+1}^{n} \mathbf{1}\big(\sigma(j)=1\big) \le a \quad \forall i \in \mathbb{N} \cap (n^{3/4}, n]\Big\}.$$

It is easy to verify that $A_{\ell-1} = \{\sigma_{(\ell-1)n} \in \Sigma^a_{n,1}\}$. We rewrite (11) as

$$\begin{aligned}&\Pr\left[\overline{A}_\ell \mid A_{\ell-1}\right] \\ &\le \Pr\left[\max_{1\le i\le j\le n} M_{(\ell-1)n+j} - M_{(\ell-1)n+j-i} > \frac{\varepsilon n^{3/4}}{2} \,\middle|\, A_{\ell-1}\right] \\ &\le \max_{\sigma\in\Sigma^a_{n,1}} \Pr\left[\max_{1\le i\le j\le n} M_{(\ell-1)n+j} - M_{(\ell-1)n+j-i} > \frac{\varepsilon n^{3/4}}{2} \,\middle|\, \sigma_{(\ell-1)n} = \sigma\right].\end{aligned}$$

By the Markov property and Lemma 6, $\{M_t\}_{t\ge(\ell-1)n}$ conditioned on $\sigma_{(\ell-1)n}$ is a supermartingale with $|M_{t+1} - M_t| \le 1$. Lemma 7 implies that

$$\Pr\left[\overline{A}_\ell \mid A_{\ell-1}\right] \le 2e^{-\frac{(\varepsilon n^{3/4})^2}{128n}} = 2e^{-\frac{\varepsilon^2\sqrt{n}}{128}},$$

which yields the result. ◀

We are now ready to prove Lemma 2.

Proof of Lemma 2. By Lemma 8 and 9, for any $\hat{\rho} > 0$, there exist an integer k and $\varepsilon > 0$ such that

$$\Pr\left[\max_{n^{3/4}<i\le n} Y^{(i)}_{kn} > \frac{1}{q} + \hat{\rho}\right] \le 2ke^{-\frac{\varepsilon^2\sqrt{n}}{128}}.$$

Recall that $Y^{(i)}_{kn}$ denotes the fraction of vertices assigned the spin 1, but this bound holds by symmetry for all spins in Q. So, setting

$$\Sigma^{\rho+}_n = \Big\{\sigma \in \Sigma_n : \frac{1}{i}\sum_{j=n-i+1}^{n} \mathbf{1}\big(\sigma(j)=\kappa\big) \le \frac{1}{q} + \rho \quad \forall i \in \mathbb{N} \cap (n^{3/4}, n] \text{ and } \forall \kappa \in Q\Big\},$$

a union bound over the spins implies

$$\Pr\left[\sigma_{kn} \notin \Sigma^{\hat{\rho}q}_n\right] \le \Pr\left[\sigma_{kn} \notin \Sigma^{\hat{\rho}+}_n\right] \le q \Pr\left[\max_{n^{3/4}<i\le n} Y^{(i)}_{kn} > \frac{1}{q} + \hat{\rho}\right] \le 2qke^{-\frac{\varepsilon^2\sqrt{n}}{128}};$$

taking $\rho = \hat{\rho}q$ establishes (3).

To prove (4), observe that the initial configuration $\sigma_0 \in \Sigma^{\rho}_n \subseteq \Sigma^{\rho+}_n$. Then, as in (10), for any integer $k > 0$

$$\Pr\left[\sigma_{kn} \notin \Sigma^{\rho+}_n\right] \le \sum_{\ell=0}^{k-1} \Pr\left[\sigma_{(\ell+1)n} \notin \Sigma^{\rho+}_n \mid \sigma_{\ell n} \in \Sigma^{\rho+}_n\right].$$

Given that $\sigma_{\ell n} \in \Sigma_n^{\rho+}$, the configuration at the next step can leave the set $\Sigma_n^{\rho+}$ only if the fluctuation is large. In particular, we obtain from Lemma 5 and a union bound over the spins that

$$\Pr\left[\sigma_{kn} \notin \Sigma_n^{\rho+}\right] \le q \sum_{\ell=0}^{k-1} \Pr\left[\max_{1\le j'<j\le n} M_{\ell n+j} - M_{\ell n+j-j'} > \frac{\varepsilon n^{3/4}}{2} \,\middle|\, \sigma_{\ell n} \in \Sigma_n^{\rho+}\right].$$

Recall that $\{M_t\}_{t\ge \ell n}$ is a supermartingale (see Lemma 6), and thus the Maximal-Azuma inequality in Lemma 7 yields that

$$\Pr\left[\sigma_{kn} \notin \Sigma_n^{\rho+}\right] \le 2qke^{-\frac{(\varepsilon n^{3/4})^2}{128n}} = 2qke^{-\frac{\varepsilon^2\sqrt{n}}{128}}.$$

Using this bound, we obtain for any integer $\hat{k} > 0$

$$\begin{aligned}
&\Pr\left[\max_{1\le i\le n} \left\|\alpha(\sigma_{(\hat{k}-1)n+i}) - \hat{e}\right\|_\infty > \rho q\right] \\
&\le \Pr\left[\max_{1\le i\le n} \left\| \alpha(\sigma_{(\hat{k}-1)n+i}) - \hat{e}\right\|_\infty > \rho q \;\middle|\; \sigma_{(\hat{k}-1)n} \in \Sigma_n^{\rho+}\right] + \Pr\left[\sigma_{(\hat{k}-1)n} \notin \Sigma_n^{\rho+}\right] \\
&\le q \Pr\left[\max_{1\le i\le n} \alpha_1\big(\sigma_{(\hat{k}-1)n+i}\big) - 1/q > \rho + \frac{1}{n^{1/4}} \;\middle|\; \sigma_{(\hat{k}-1)n} \in \Sigma_n^{\rho+}\right] + 2q(\hat{k}-1)e^{-\frac{\varepsilon^2\sqrt{n}}{128}} \\
&\le q \max_{\sigma\in\Sigma_n^{\rho+}} \Pr\left[\max_{1\le i\le n} \alpha_1\big(\sigma_{(\hat{k}-1)n+i}\big) - 1/q > \rho + \frac{1}{n^{1/4}} \;\middle|\; \sigma_{(\hat{k}-1)n} = \sigma\right] + 2q(\hat{k}-1)e^{-\frac{\varepsilon^2\sqrt{n}}{128}},
\end{aligned} \tag{12}$$

where the second inequality is obtained by taking union bound over each spin. We consider the following events,

$$\begin{aligned}
E_{\hat{k}} &= \left\{\max_{n^{3/4}<i\le n} Y^{(i)}_{(\hat{k}-1)n} \le \frac{1}{q} + \rho\right\}, \\
B_{\hat{k}} &= \left\{\max_{1\le i\le j\le n} D^{(i)}_{(\hat{k}-1)n+j} < \frac{\varepsilon n^{3/4}}{2}\right\}, \text{ and} \\
C_{\hat{k}} &= \left\{\max_{1\le i\le n} Y_{(\hat{k}-1)n+i} \le \frac{1}{q} + \rho + \frac{1}{n^{1/4}}\right\}.
\end{aligned}$$

Lemma 5 implies that $\overline{C}_{\hat{k}} \subseteq \overline{E}_{\hat{k}} \cup \overline{B}_{\hat{k}}$. We recall that $D_t^\ell = M_t - M_{t-\ell}$ for any positive integer t and integer ℓ such that $1 \le \ell \le \min\{t, n\}$. Using the fact $\left\{\sigma_{(\hat{k}-1)n} \in \Sigma_n^{\rho+}\right\} \subseteq \overline{E}_{\hat{k}}$, we obtain for any $\sigma \in \Sigma_n^{\rho+}$

$$\begin{aligned}
&\Pr\left[\max_{1\le i\le n} \alpha_1\big(\sigma_{(\hat{k}-1)n+i}\big) - 1/q > \rho + \frac{1}{n^{1/4}} \;\middle|\; \sigma_{(\hat{k}-1)n} = \sigma\right] = \Pr\left[\overline{C}_{\hat{k}} \,\middle|\, \sigma_{(\hat{k}-1)n} = \sigma\right] \\
&\quad\le \Pr\left[\overline{E}_{\hat{k}} \,\middle|\, \sigma_{(\hat{k}-1)n} = \sigma\right] + \Pr\left[\overline{B}_{\hat{k}} \,\middle|\, \sigma_{(\hat{k}-1)n} = \sigma\right] \\
&\quad\le \Pr\left[\max_{1\le i<j\le n} M_{(\hat{k}-1)n+j} - M_{(\hat{k}-1)n+j-i} > \frac{\varepsilon n^{3/4}}{2} \Big| \sigma_{(\hat{k}-1)n} = \sigma\right] \\
&\quad\le 2e^{-\frac{\varepsilon^2\sqrt{n}}{128}},
\end{aligned}$$

where we apply Lemmas 6 and 7 in the last inequality. Plugging this bound into (12) yields the result. ◀

2.4 Proofs of auxiliary Facts

In this section, we complete the proof of Lemma 2 by providing the proofs of Lemmas 6 and 8. Let us introduce some notation first. Let

$$\mathcal{S}_q = \Big\{x \in \mathbb{R}^q_{\geq 0} : \sum_{i=1}^{q} x_i = 1\Big\},$$

$$\tilde{\mathcal{S}}_q = \Big\{x \in \mathbb{R}^q_{\geq 0} : \sum_{i=1}^{q} x_i = 1 - \frac{1}{n}\Big\}, \text{ and}$$

$$\mathcal{S}'_q = \Big\{x \in \mathbb{R}^q_{+} : \sum_{j=1}^{q} x_j = 1 \text{ and } x_i \geq \frac{1}{e^{\beta} + (q-1)} \quad \forall i \in Q\Big\}.$$

We define the function $g_\beta : \mathcal{S}_q \cup \tilde{\mathcal{S}}_q \to \mathcal{S}'_q$, as $g_\beta(s) = \big(g^1_\beta(s), \cdots, g^q_\beta(s)\big)$ where

$$g^j_\beta(s) = \frac{e^{\beta s_j}}{\sum_{i=1}^{q} e^{\beta s_i}}.$$

For $i, j \in Q$, let $g^{i\to j}_\beta : \mathcal{S}_q \to \mathcal{S}'_q$ be given by

$$g^{i\to j}_\beta(s) = g^j_\beta\Big(s - \frac{1}{n}e_i\Big),$$

where recall that $e_i \in \{0,1\}^q$ is the indicator vector with 1 in the i-th coordinate and 0 elsewhere. This function corresponds to the probability that a site update flips the vertex from spin i to j from configuration with proportion vector s. We establish the following inequality involving $g^{i\to j}_\beta$ and $G_{\beta,n}$; its proof is given in the full version of the paper [5].

▶ **Lemma 10.** *For $s \in \mathcal{S}_q$ and $i, j \in Q$, we have*

$$g^{i\to j}_\beta(s) \leq g^j_\beta(s) + \frac{\beta}{n} \leq G_{\beta,n}(s_j).$$

We can now provide the proof of Lemma 6.

Proof of Lemma 6. Let $t \geq 1$ be a integer and let j be the spin of the vertex to be update updated at time $t+1$. Then,

$$\begin{aligned}\mathbb{E}[M_{t+1} - M_t|\mathcal{F}_t] &= \mathbb{E}[X_{t+1} - G_{\beta,n}(Y_t)|\mathcal{F}_t] = \mathbb{E}[X_{t+1}|\mathcal{F}_t] - G_{\beta,n}(Y_t)\\ &= g^{j\to 1}_\beta\big(\alpha(\sigma_t)\big) - G_{\beta,n}(Y_t) \leq 0,\end{aligned}$$

where the last inequality follows from Lemma 10. ◀

3 Spin fraction analysis

In this section, we begin our analysis of Phase 2 of the coupling. In particular, we analyze rate of convergence of the systematic scan to a $O(1/\sqrt{n})$ neighborhood of $\hat{e}$ from a configuration $\sigma_0 \in \Sigma^\rho_n$. We start with the following bounds for the function g^ℓ_β which follow from its Taylor expansion; their proofs are provided in the full version of the paper [5].

▶ **Lemma 11.** *For any $\ell \in Q$ and $s, \tilde{s} \in \mathcal{S}_q$,*

(i) *$\exists\ R : \mathcal{S}_q \to \mathbb{R}$ such that $g^\ell_\beta(s) = \frac{1}{q} + \frac{\beta}{q}\Big(s_\ell - \frac{1}{q}\Big) + R(s)$ and $|R(s)| \leq 3q\beta^2\|s - \hat{e}\|_2^2$;*

(ii) *$\big\|g_\beta(s) - g_\beta(\tilde{s})\big\|_1 = \frac{\beta}{q}\big\|s - \tilde{s}\big\|_1 + \|s - \tilde{s}\|_1\Big\{O\big(\|s - \hat{e}\|_1 + \|\tilde{s} - \hat{e}\|_1\big)\Big\}$.*

Now, for a vector $X = (X_1, \dots, X_q) \in \mathbb{R}^q$, let

$$\mathrm{Var}(X) = \mathbb{E}\big[\|X - \mathbb{E}X\|_2^2\big] = \sum_{i=1}^{q} \mathbb{E}\big[\big(X_i - \mathbb{E}X_i\big)^2\big] = \sum_{i=1}^{q} \mathrm{Var}(X_i).$$

We prove the following bound on the variance of the q-dimensional process $\alpha(\sigma_t)$, assuming that $\sigma_0 \in \Sigma_n^\rho$; its proof is provided in the full version of the paper [5].

▶ **Lemma 12.** *There exists a constant $\rho = \rho(\beta, q) > 0$ such that for all $0 \le t \le n\sqrt{n}$ and $\sigma_0 \in \Sigma_n^\rho$, we have $\mathrm{Var}\big(\alpha(\sigma_t)\big) = O\big(1/n\big)$.*

We also require a tight control on the following recurrence; its analysis is fleshed out in Section 4.

▶ **Lemma 13.** *Let $\{x_t = (x_t^{(1)}, \dots, x_t^{(q)})\}_{t \ge 0}$ be a sequence of vectors in $\mathbb{R}^q_{\ge 0}$ such that for all $j \in Q$ and $t \ge n$, we have the recursion*

$$x_t^{(j)} = \frac{\beta}{nq} \sum_{i=t-n}^{t-1} x_i^{(j)} + \frac{C}{n} \sum_{i=t-n}^{t-1} \|x_i\|_2^2 + \frac{C}{n},$$

where C is a positive constant. Then, if for all $i \in \{0, \dots, n-1\}$ and all $j \in Q$, we have $|x_i^{(j)}| \le \rho$ for a small enough constant $\rho > 0$, then there exists constant $\hat{C} > 0$ such that $\|x_t\|_2^2 \le \hat{C}\rho^2\gamma_n^{2t} + \hat{C}/n$, where γ_n is the unique positive real root of the equation $\sum_{i=1}^{n} \gamma_n^{-i} = nq/\beta$.

We combine the three lemmas above to obtain the correct rate of contraction towards $\hat{e}$ of the proportions vector under the systematic scan dynamics. Let

$$u_t = \begin{cases} e_{J_t} & \text{if } t \ge 1 \\ e_{\sigma_0(n+t)} & \text{if } -n < t \le 0, \end{cases}$$

where recall that J_t denotes the random variable corresponding to the spin chosen in the t-th site update. With this notation, we have

$$\alpha(\sigma_t) = \frac{1}{n} \sum_{i=t-n+1}^{t} u_i. \tag{13}$$

▶ **Lemma 14.** *Let γ_n be the unique positive real root of the equation $\sum_{i=1}^{n} \gamma_n^{-i} = nq/\beta$. Then, for all integer $t \ge 0$ such that $t \le n\sqrt{n}$ there exist constants $\rho = \rho(\beta, q)$ and $C = C(\rho, \beta, q)$ such that for any $\sigma_0 \in \Sigma_n^\rho$*

$$\mathbb{E}\big[\|\alpha(\sigma_t) - \hat{e}\|_2^2\big] \le C\rho^2\gamma_n^{2t} + \frac{C}{n}.$$

Proof. We let $t_m = t_m(t) = \max\{1, t - n + 1\}$. From (13), for all $t \ge 0$ and $\ell \in Q$, we have

$$\alpha_\ell(\sigma_t) = \frac{1}{n} \sum_{i=t-n+1}^{0} \mathbf{1}\big(\sigma_0(n+i) = \ell\big) + \frac{1}{n} \sum_{i=t_m}^{t} \mathbf{1}\big(J_i = \ell\big),$$

and thus

$$\begin{aligned} \mathbb{E}\big[\alpha_\ell(\sigma_t)\big] &= \frac{1}{n} \sum_{i=t-n+1}^{0} \mathbf{1}\big(\sigma_0(n+i) = \ell\big) + \frac{1}{n} \sum_{i=t_m}^{t} \mathbb{E}\big[\mathbf{1}(J_i = \ell)\big] \\ &= \frac{1}{n} \sum_{i=t-n+1}^{0} \mathbf{1}\big(\sigma_0(n+i) = \ell\big) + \frac{1}{n} \sum_{i=t_m}^{t} \mathbb{E}\Big[g_\beta^\ell\Big(\alpha(\sigma_{i-1}) - \frac{1}{n}u_{i-n}\Big)\Big]. \end{aligned} \tag{14}$$

From Lemmas 10 and 11(i), we obtain

$$\begin{aligned}\mathbb{E}\Big[g_\beta^\ell\Big(\alpha(\sigma_{i-1})-\frac{1}{n}u_{i-n}\Big)\Big] \le& \frac{\beta}{n}+\mathbb{E}\Big[g_\beta^\ell(\alpha(\sigma_{i-1}))\Big]\\ \le& \frac{\beta}{n}+\frac{1}{q}+\frac{\beta}{q}\mathbb{E}\Big[\alpha_\ell(\sigma_{i-1})-\frac{1}{q}\Big]+3q\beta^2\mathbb{E}\big[\|\alpha(\sigma_i)-\hat{e}\|_2^2\big]\\ \le& \frac{\beta}{n}+3q\beta^2\mathrm{Var}\big(\alpha(\sigma_i)\big)+\frac{1}{q}+\frac{\beta}{q}\mathbb{E}\Big[\alpha_\ell(\sigma_{i-1})-\frac{1}{q}\Big]\\ &+3q\beta^2\big\|\mathbb{E}[\alpha(\sigma_i)-\hat{e}]\big\|_2^2.\end{aligned}$$

By Lemma 12, there exists a constant $C_1 > 0$ such that the following holds for all $t \in \{0,\cdots,n\sqrt{n}\}$

$$\mathbb{E}\Big[g_\beta^\ell\Big(\alpha(\sigma_{i-1})-\frac{1}{n}u_{i-n}\Big)\Big] \le \frac{C_1}{n}+\frac{1}{q}+\frac{\beta}{q}\mathbb{E}\Big[\alpha_\ell(\sigma_{i-1})-\frac{1}{q}\Big]+3q\beta^2\big\|\mathbb{E}[\alpha(\sigma_i)-\hat{e}]\big\|_2^2.$$

Plugging the above inequality into (14)

$$\begin{aligned}\mathbb{E}\big[\alpha_\ell(\sigma_t)\big] \le& \frac{C_1}{n}+\frac{1}{n}\sum_{i=t-n+1}^{0}\Big[\mathbf{1}\big(\sigma_0(n+i)=\ell\big)\Big]\\ &+\frac{t-t_m+1}{nq}+\frac{1}{n}\sum_{i=t_m}^{t}\frac{\beta}{q}\mathbb{E}\Big[\alpha_\ell(\sigma_{i-1})-\frac{1}{q}\Big]+\frac{3q\beta^2}{n}\sum_{i=t_m}^{t}\big\|\mathbb{E}[\alpha(\sigma_{i-1})-\hat{e}]\big\|_2^2,\end{aligned}$$

and subtracting $1/q$ from both sides yields,

$$\begin{aligned}\mathbb{E}\Big[\alpha_\ell(\sigma_t)-\frac{1}{q}\Big] \le& \frac{C_1}{n}+\frac{1}{n}\sum_{i=t-n+1}^{0}\Big[\mathbf{1}\big(\sigma_0(n+i)=\ell\big)-\frac{1}{q}\Big]\\ &+\frac{1}{n}\sum_{i=t_m}^{t}\frac{\beta}{q}\mathbb{E}\Big[\alpha_\ell(\sigma_{i-1})-\frac{1}{q}\Big]+\frac{3q\beta^2}{n}\sum_{i=t_m}^{t}\big\|\mathbb{E}[\alpha(\sigma_{i-1})-\hat{e}]\big\|_2^2. \qquad (15)\end{aligned}$$

Let $\{x_t\}_{t\ge 0}$ be a sequence of vectors in $\mathbb{R}^q$ such that for all $j \in Q$ and $t \ge n$, we have the recursion

$$x_t^{(j)} = \frac{\beta}{nq}\sum_{i=t-n}^{t-1}x_i^{(j)}+\frac{C_2}{n}\sum_{i=t-n}^{t-1}\|x_i\|_2^2+\frac{C_2}{n}.$$

where $C_2 = \max\{C_1, 3q\beta^2\}$. For all $i \in \{0,\dots,n-1\}$ and all $j \in Q$, $x_i^{(j)} = 2\rho$. We prove by induction on t that for every $\ell \in Q$ and $t \in \{0,\cdots,n\sqrt{n}\}$, the following holds

$$\Big|\mathbb{E}\Big[\alpha_\ell(\sigma_t)-\frac{1}{q}\Big]\Big| \le x_t^{(\ell)}. \qquad (16)$$

For $t = 0$, the claim trivially holds since $\sigma_0 \in \Sigma_n^\rho$ and therefore $|\alpha_\ell(\sigma_0)-1/q| \le \rho < 2\rho$. We consider three cases for the inductive step. We assume first that the claim holds for all $t \in \{1,\cdots,\kappa-1\}$ for some $\kappa \in \{1,\cdots,n-n^{3/4}-1\}$. For $t = \kappa$, we obtain from (15) and the triangle inequality that

$$\begin{aligned}\Big|\mathbb{E}\Big[\alpha_\ell(\sigma_t)-\frac{1}{q}\Big]\Big| \le& \frac{C_1}{n}+\frac{1}{n}\Big|\sum_{i=t-n+1}^{0}\Big[\mathbf{1}\big(\sigma_0(n+i)=\ell\big)-\frac{1}{q}\Big]\Big|\\ &+\frac{\beta}{nq}\sum_{i=t_m}^{t}\Big|\mathbb{E}\Big[\alpha_\ell(\sigma_{i-1})-\frac{1}{q}\Big]\Big|+\frac{3q\beta^2}{n}\sum_{i=t_m}^{t}\big\|\mathbb{E}[\alpha(\sigma_{i-1})-\hat{e}]\big\|_2^2.\end{aligned}$$

Since $\sigma_0 \in \Sigma_n^\rho$, we observe that $\left|\sum_{i=t-n+1}^{0}\left[\mathbf{1}\big(\sigma_0(n+i)=\ell\big)-\frac{1}{q}\right]\right| \le (n-t)\rho$, using the inductive hypothesis to bound the second and third sums we obtain

$$\begin{aligned}\left|\mathbb{E}\Big[\alpha_\ell(\sigma_t)-\frac{1}{q}\Big]\right| &\le \frac{C_1}{n}+\frac{(n-t)\rho}{n}+\frac{\beta}{nq}(2t\rho)+\frac{12q\beta^2}{n}\big(tq\rho^2\big)\\ &= \frac{C_1}{n}+\rho\Big[1+\frac{t}{n}\Big(\big(2\beta/q\big)-1+12q^2\beta^2\rho\Big)\Big].\end{aligned}$$

Set $1-\beta/q=\delta$ so that $2\beta/q-1=1-2\delta$ (recall that $\beta/q<1$). Since $12q^2\beta^2$ is a fixed constant, we can choose ρ small enough so that $12q^2\beta^2\rho<\delta$. Plugging these into the above inequality:

$$\left|\mathbb{E}\Big[\alpha_\ell(\sigma_t)-\frac{1}{q}\Big]\right| \le \frac{C_1}{n}+\rho\Big[1+(1-\delta)\Big] = 2\rho+\Big(\frac{C_1}{n}+-\rho\delta\Big) \le 2\rho \le x_t^{(\ell)}$$

provided n is large enough. For the second case, let us assume that the claim holds for all $t\in\{1,\cdots,\kappa-1\}$ for some $\kappa\in\{n-n^{3/4},\cdots,n-1\}$. For $t=\kappa$:

$$\begin{aligned}\left|\mathbb{E}\Big[\alpha_\ell(\sigma_t)-\frac{1}{q}\Big]\right| &\le \frac{C_1}{n}+\frac{n^{3/4}}{n}+\frac{2\beta t\rho}{nq}+\frac{12q^2\beta^2t\rho^2}{n} \le \frac{C_1}{n}+\frac{1}{n^{1/4}}+\rho\Big(\frac{2\beta}{q}+12q^2\beta^2\rho\Big)\\ &\le 2\rho+\Big(\frac{C_1}{n}+\frac{1}{n^{1/4}}-\rho\delta\Big) \le 2\rho \le x_t^{(\ell)}.\end{aligned}$$

For the third case, We assume that the claim holds for all $t\in\{0,\cdots,\kappa-1\}$ for some $\kappa\in\{n,\cdots,n\sqrt{n}\}$. For $t=\kappa$,

$$\begin{aligned}\left|\mathbb{E}\Big[\alpha_\ell(\sigma_t)-\frac{1}{q}\Big]\right| &\le \frac{C_1}{n}+\frac{\beta}{qn}\left|\sum_{i=t-n}^{t-1}\frac{\beta}{q}\mathbb{E}\Big[\alpha_\ell(\sigma_i)-\frac{1}{q}\Big]\right|+\frac{3q\beta^2}{n}\sum_{i=t-n}^{t-1}\big\|\mathbb{E}[\alpha(\sigma_i)-\hat{e}]\big\|_2^2\\ &\le \frac{C_2}{n}+\frac{\beta}{qn}\sum_{i=t-n}^{t-1}x_i^{(\ell)}+\frac{C_2}{n}\sum_{i=t-n}^{t-1}\|x_i\|_2^2 = x_t^{(\ell)},\end{aligned}$$

which proves the claim (16). Using the Lemma 13, we show that the following holds for some constant $C_3>0$

$$\Big\|\mathbb{E}\big[\alpha(\sigma_t)-\hat{e}\big]\Big\|_2^2 \le \|x_t\|_2^2 \le C_3\rho^2\gamma_n^{2t}+\frac{C_3}{n}.$$

Using Lemma 12 and the above equation, we prove this lemma. ◀

4 Recursion

In this section, we prove Lemma 13. For integer $n\ge 1$, we define $\gamma_n\in(0,1)$ as the unique positive root of

$$\frac{\beta}{nq}\sum_{i=1}^{n}\gamma_n^{-i}=1. \tag{17}$$

The left hand side of this equation is continuously decreasing in $\gamma_n\in(0,\infty)$, and it is $\frac{\beta}{q}$ when evaluated at $\gamma_n=1$. For $\gamma_n=\beta/q$, the left hand side is

$$\frac{\beta}{nq}\sum_{i=1}^{n}\Big(\frac{q}{\beta}\Big)^i=\frac{1}{n}\sum_{i=1}^{n}\Big(1+\frac{q}{\beta}+\cdots+\frac{q^{n-1}}{\beta^{n-1}}\Big)>1,$$

where the last inequality follows from $\frac{\beta}{q}<1$. There is indeed a unique positive solution for (17) in $(\beta/q,1)$.

We will show that the unique positive solution of (17) will be asymptotically approximated by $1 - b/n$ where b is the positive real root of the following equation

$$\frac{\beta(1-e^{-b})}{qb} - e^{-b} = 0, \tag{18}$$

while b does not have an elementary closed form, it can be expressed as

$$b = -\frac{\beta}{q} - W_{-1}\left(-\frac{\beta}{q}\exp\left(-\beta/q\right)\right),$$

where W_{-1} is the negative branch of the Lambert W function. Formally, for real numbers x and y, the equation $ye^y = x$ has real roots for y if $-\frac{1}{e} \leq x < 0$ and one of its real roots is $W_{-1}(x)$. We establish the following approximation for the root of (17); its proof is provided in the full version [5].

▶ **Lemma 15.** *Let $\hat{\gamma}_n = 1 - b/n$. Then, $|\gamma_n - \hat{\gamma}_n| = O(n^{-2})$ and for every $t \in [0, \ldots, n\sqrt{n}]$, we have $e^{-bt/n}/2 \leq \gamma_n^t \leq 2e^{-bt/n}$.*

A direct consequence of Lemma 15 is that

$$\sum_{i=0}^{\infty} \gamma_n^i = \Theta(n), \quad \text{and} \tag{19}$$

$$\gamma_n^j = \Theta(1) \text{ for all } 0 \leq j \leq n. \tag{20}$$

We use γ_n to bound the recurrence in the following lemma, which we later use to prove Lemma 13.

▶ **Lemma 16.** *Let $\{b_t\}_{t\geq n}$ be a sequence of nonnegative real numbers and define $\{d_t\}_{t\geq 0}$ as*

$$d_t = \begin{cases} 0 & \text{for } 0 \leq t < n, \\ b_t + \frac{\beta}{nq}\sum_{i=t-n}^{t-1} d_i & \text{for } t \geq n. \end{cases}$$

Then, there exists a constant $C \geq 1$ such that for all $t \geq n$, we have $d_t \leq b_t + \frac{C\gamma_n^t}{n}\sum_{i=n}^{t} b_i\gamma_n^{-i}$.

The proof of this lemma is deferred to the full version of the paper [5]. We are now ready to provide the proof of Lemma 13.

Proof of Lemma 13. Fix $k \in Q$ and consider the sequence $\{y_t\}_{t\geq 0}$ defined as $y_i = x_i^{(k)}$ for $i \in \{0, \ldots, n-1\}$ and $y_t = \frac{1}{n}\frac{\beta}{q}\sum_{i=t-n}^{t-1} y_i$ when $t \geq n$. Let $d_t = x_t^{(k)} - y_t$ for all $t \geq 0$. Observe that $d_t = 0$ for all $0 \leq t < n$. For $t \geq n$, we have

$$\begin{aligned} d_t = x_t^{(k)} - y_t &= \frac{\beta}{nq}\sum_{i=t-n}^{t-1}\left(x_i^{(k)} - y_i\right) + \frac{C}{n}\sum_{i=t-n}^{t-1}\|x_i\|_2^2 + \frac{C}{n} \\ &= \frac{\beta}{nq}\sum_{i=t-n}^{t-1} d_i + \frac{C}{n}\sum_{i=t-n}^{t-1}\|x_i\|_2^2 + \frac{C}{n} = b_t + \frac{\beta}{nq}\sum_{i=t-n}^{t-1} d_i, \end{aligned}$$

where we let $b_t = \frac{C}{n}\sum_{i=t-n}^{t-1}\|x_i\|_2^2 + \frac{C}{n}$ for all $t \geq n$. From Lemma 16, there exists $C_1 > 1$ such that the following holds for all $t \geq n$

$$d_t \leq b_t + \frac{C_1\gamma_n^t}{n}\sum_{i=n}^{t}\gamma_n^{-i}b_i.$$

For convenience, we choose the constant C_1 large enough such that also $\gamma_n^{-n} \leq C_1$ and

$$\sum_{i=0}^{\infty} \gamma_n^i \leq C_1 n; \tag{21}$$

see (20) and (19). We expand d_t as follows

$$d_t \leq \frac{C}{n} \sum_{i=t-n}^{t-1} \|x_i\|_2^2 + \frac{C}{n} + \frac{CC_1\gamma_n^t}{n^2} \sum_{i=n}^{t} \gamma_n^{-i} \sum_{j=i-n}^{i-1} \|x_j\|_2^2 + \frac{CC_1}{n^2} \sum_{i=n}^{t} \gamma_n^{t-i}.$$

From (21), we see $\sum_{i=n}^{t} \gamma_n^{t-i} \leq C_1 n$ and plugging this in the above bound, we obtain

$$\begin{aligned} d_t &\leq \frac{C}{n} \sum_{i=t-n}^{t-1} \|x_i\|_2^2 + \frac{C}{n} + \frac{CC_1\gamma_n^t}{n^2} \sum_{i=n}^{t} \gamma_n^{-i} \sum_{j=i-n}^{i-1} \|x_j\|_2^2 + \frac{CC_1^2}{n} \\ &\leq \frac{2CC_1^2}{n} + \frac{C}{n} \sum_{i=t-n}^{t-1} \|x_i\|_2^2 + \frac{CC_1\gamma_n^t}{n^2} \sum_{i=n}^{t} \gamma_n^{-i} \sum_{j=i-n}^{i-1} \|x_j\|_2^2 \end{aligned} \tag{22}$$

where we use the fact that $C + CC_1^2 \leq 2CC_1^2$. Observe also that

$$\begin{aligned} \sum_{i=n}^{t} \gamma_n^{-i} \sum_{j=i-n}^{i-1} \|x_j\|_2^2 &= \sum_{j=0}^{n-1} \gamma_n^{j-n} \sum_{i=n}^{t} \gamma_n^{-(i+j-n)} \|x_{i+j-n}\|_2^2 \\ &\leq \gamma_n^{-n} \sum_{j=0}^{n-1} \sum_{i=n}^{t} \gamma_n^{-(i+j-n)} \|x_{i+j-n}\|_2^2 \leq nC_1 \sum_{i=0}^{t-1} \gamma_n^{-i} \|x_i\|_2^2, \end{aligned} \tag{23}$$

where the last inequality follows from $\gamma_n^{-n} \leq C_1$. Plugging (23) into (22), we obtain

$$\begin{aligned} d_t &\leq \frac{2CC_1^2}{n} + \frac{C}{n} \sum_{i=t-n}^{t-1} \|x_i\|_2^2 + \frac{CC_1^2}{n} \gamma_n^t \sum_{i=0}^{t-1} \gamma_n^{-i} \|x_i\|_2^2 \\ &\leq \frac{2CC_1^2}{n} + \frac{C\gamma_n^{-n}}{n} \sum_{i=t-n}^{t-1} \gamma_n^{t-i} \|x_i\|_2^2 + \frac{CC_1^2}{n} \gamma_n^t \sum_{i=0}^{t-1} \gamma_n^{-i} \|x_i\|_2^2 \\ &\leq \frac{2CC_1^2}{n} + \frac{C(C_1+1)C_1}{n} \gamma_n^t \sum_{i=0}^{t-1} \gamma_n^{-i} \|x_i\|_2^2 \leq \frac{2CC_1^2}{n} + \frac{2CC_1^2}{n} \gamma_n^t \sum_{i=0}^{t-1} \gamma_n^{-i} \|x_i\|_2^2. \end{aligned} \tag{24}$$

We show next by induction on t that the following holds for all $t \geq 0$

$$y_t \leq C_1 \rho \gamma_n^t. \tag{25}$$

For all $0 \leq t < n$, we have $C_1\rho\gamma_n^t \geq \rho\gamma_n^{-n}\gamma_n^t = \rho\gamma_n^{t-n} \geq \rho \geq y_t$. Therefore, the base case holds. We assume that the claim holds for all $t < \ell$ where $\ell \geq n$. For $t = \ell$,

$$y_t = \frac{\beta}{nq} \sum_{i=t-n}^{t-1} y_i \leq \frac{\beta}{nq} \sum_{i=t-n}^{t-1} C_1\rho\gamma_n^i = C_1\rho \frac{\beta}{nq} \sum_{i=t-n}^{t-1} \gamma_n^i = C_1\rho\gamma_n^t \frac{\beta}{nq} \sum_{i=0}^{n-1} \gamma_n^{i-n} = C_1\rho\gamma_n^t,$$

which proves the claim in (25). Now, we show the following using induction

$$\|x_t\|_2^2 < 4C_1^2 q\rho^2\gamma_n^{2t} + \frac{4C_1^2 q}{n}. \tag{26}$$

We obtain the following for all t such that $0 \le t < n$,

$$4C_1^2 q\rho^2\gamma_n^{2t} + \frac{4C_1^2 q}{n} \ge 4C_1^2 q\rho^2\gamma_n^{2n} \ge 4\gamma_n^{-2n} q\rho^2\gamma_n^{2n} \ge 4q\rho^2 \ge \|x_t\|_2^2$$

Therefore, the base case holds. We assume that the claim holds for all $t < \ell$ where $\ell \ge n$. For $t = \ell$, we obtain the following from (25) and (24),

$$|x_t^{(k)}| \le |y_t| + |d_t| \le C_1\rho\gamma_n^t + \frac{2CC_1^2\gamma_n^t}{n}\sum_{i=0}^{t-1}\gamma_n^{-i}\|x_i\|_2^2 + \frac{2CC_1^2}{n}. \tag{27}$$

The induction hypothesis then implies that

$$\begin{aligned}\gamma_n^t\sum_{i=0}^{t-1}\gamma_n^{-i}\|x_i\|_2^2 &\le \gamma_n^t\sum_{i=0}^{t-1}\gamma_n^{-i}\Big(4C_1^2 q\rho^2\gamma_n^{2i} + \frac{4C_1^2 q}{n}\Big) \le 4C_1^2 q\gamma_n^t\sum_{i=0}^{t-1}\gamma_n^{-i}\big(\rho^2\gamma_n^{2i} + 1/n\big)\\ &\le 4C_1^2 q\Big(\rho^2\gamma_n^t\sum_{i=0}^{t-1}\gamma_n^i + \frac{1}{n}\sum_{i=0}^{t-1}\gamma_n^{t-i}\Big) \le 4C_1^2 q\big(\rho^2\gamma_n^t C_1 n + C_1\big)\end{aligned}$$

We then plug the above bound into (27) to obtain

$$\begin{aligned}|x_t^{(k)}| &\le C_1\rho\gamma_n^t + \frac{2CC_1^2}{n}\Big(4C_1^2 q\big(\rho^2\gamma_n^t C_1 n + C_1\big)\Big) + \frac{2CC_1^2}{n}\\ &\le C_1\rho\gamma_n^t + 8CC_1^5\rho^2 q\gamma_n^t + \frac{8CC_1^5 q + 2CC_1^2}{n}\\ &\le \rho\gamma_n^t C_1\big(1 + 8CC_1^4\rho q\big) + \frac{8CC_1^5 q + 2CC_1^2}{n}.\end{aligned}$$

Let $C_2 = C_1\big(1+8CC_1^4\rho q\big)$ and $C_3 = 8CC_1^5 q + 2CC_1^2$ for ease of notation. Since k is arbitrary, the above inequality holds for all coordinates in Q, hence,

$$\|x_t\|_2^2 \le 2q\rho^2\gamma_n^{2t}C_2^2 + \frac{2qC_3^2}{n^2},$$

where we use the fact that $(w+z)^2 \le 2w^2 + 2z^2$ for $w, z \in \mathbb{R}$. We want to show $2q\rho^2\gamma_n^{2t}C_2^2 + \frac{2qC_3^2}{n^2} \le 4C_1^2 q\rho^2\gamma_n^{2t} + \frac{4C_1^2 q}{n}$ by equating the coefficients of γ_n^{2t} and $1/n$. Comparing the coefficient of γ_n^{2t}, we show $C_2 \le \sqrt{2}C_1$ or equivalently, $1 + 8CC_1^4\rho q \le \sqrt{2}$. We choose $\rho > 0$ small enough to make the above inequality true. Moreover, $2qC_3^2/n \le 4C_1^2 q$ holds trivially. Therefore, we proved our claim (26), hence choosing $\hat{C} = 4C_1^2 q$ proves the lemma. ◀

5 Contraction of Hamming Distance

Our goal in this section is to establish Lemma 3 which ensures that **Phase 2** of our coupling succeeds with the desired ε probability. We use the following lemma to upper bound expectation of Hamming distance; its proof is deferred to the full version of the paper [5].

▶ **Lemma 17.** *Let $\{x_t\}_{t\ge 0}$ be a sequence of positive real numbers, such that $x_t \le n$ for $t < n$, and for $t \ge n$ we have $x_t = \frac{\beta}{nq}\sum_{i=t-n}^{t-1} x_i + C\sum_{i=t-n}^{t-1}\gamma_n^{2i} + C$, where $C > 0$ is a constant. Then, there exists a constant $\hat{C} > 0$ such that for all $t \ge 0$ $x_t \le \hat{C}n\gamma_n^t + \hat{C}$.*

We are now ready to provide the proof of Lemma 3.

Proof of Lemma 3. We couple two instances using the optimal single site coupling for two copies of the dynamics. Formally, we define a distribution $\nu_{\sigma,v}$ on Q such that $\nu_{\sigma,v}(j) = g_\beta^j\big(\alpha(\sigma) - \frac{1}{n}e_{\sigma(v)}\big)$ where $\sigma \in \Sigma_n$, $j \in Q$ and $v \in V$. From a pair of configurations σ_{t-1} and $\tilde{\sigma}_{t-1}$, we couple the t-th site update as follows:

1. Let $v = \big(t \bmod n\big) + 1$ be the vertex to be updated.
2. Draw spins J_t and $\tilde{J}_t$ from an optimal coupling of the distribution $\nu_{\sigma_{t-1},v}$ and $\nu_{\tilde{\sigma}_{t-1},v}$ respectively.
3. Set $\sigma_t(v) = J_t$ and $\tilde{\sigma}_t(v) = \tilde{J}_t$ leaving all other vertices unchanged.

We show next that under the optimal site coupling, there exists a constant $C_1 > 0$ such that $\forall\ t \geq n$, $\mathbb{E}[d_H(\sigma_t, \tilde{\sigma}_t)]$ is at most

$$C_1 + \frac{\beta}{nq} \sum_{i=t-n}^{t-1} \mathbb{E}[d_H(\sigma_i, \tilde{\sigma}_i)] + C_1 \sum_{i=t-n}^{t-1} \Big(\mathbb{E}\big[\|\alpha(\sigma_i) - \hat{e}\|_2^2\big] + \mathbb{E}\big[\|\alpha(\tilde{\sigma}_i) - \hat{e}\|_2^2\big] \Big). \tag{28}$$

The Hamming distance between two copies after t site updates for $t \geq n$ is $d_H(\sigma_t, \tilde{\sigma}_t) = \sum_{i=t-n+1}^{t} \mathbf{1}\left(J_i \neq \tilde{J}_i\right)$. Taking expectations on both sides we obtain for all $t \geq n$ that

$$\begin{aligned} \mathbb{E}\left[d_H(\sigma_t, \tilde{\sigma}_t)\right] = \sum_{i=t-n+1}^{t} \mathbb{E}\left[\mathbf{1}\left(J_i \neq \tilde{J}_i\right)\right] &= \sum_{i=t-n+1}^{t} \mathbb{E}\left[\mathbb{E}\left[\mathbf{1}\big(J_i \neq \tilde{J}_i\big)\big|\mathcal{F}_{i-1}\right]\right] \\ &= \sum_{i=t-n+1}^{t} \mathbb{E}\left[\Pr\left[J_i \neq \tilde{J}_i \Big| \mathcal{F}_{i-1}\right]\right]. \end{aligned} \tag{29}$$

Since we are optimally coupling each co-ordinate, the following holds,

$$\Pr\left[J_i \neq \tilde{J}_i \mid \mathcal{F}_{i-1}\right] = \frac{1}{2}\Big\| g_\beta\Big(\alpha(\sigma_i) - \frac{1}{n} e_{I_i}\Big) - g_\beta\Big(\alpha(\tilde{\sigma}_i) - \frac{1}{n} e_{\tilde{I}_i}\Big)\Big\|_1. \tag{30}$$

For any $I, \tilde{I} \in Q$, we have

$$\begin{aligned} \Big\| g_\beta(s - \frac{1}{n} e_I) - g_\beta(\tilde{s} - \frac{1}{n} e_{\tilde{I}})\Big\|_1 &= \|g_\beta(s) - g_\beta(\tilde{s})\|_1 + \frac{2q^2\beta}{n} \\ &\leq \frac{\beta}{q}\|s - \tilde{s}\|_1 + O\big((\|s - \hat{e}\|_1 + \|\tilde{s} - \hat{e}\|_1)^2\big) + \frac{2q^2\beta}{n} \\ &\leq \frac{\beta}{q}\|s - \tilde{s}\|_1 + O\big(\|s - \hat{e}\|_2^2 + \|\tilde{s} - \hat{e}\|_2^2\big) + \frac{2q^2\beta}{n}, \end{aligned} \tag{31}$$

where the first inequality follows from Lemma 11(ii) and the fact that $\|s - \tilde{s}\|_1 \leq \|s - \hat{e}\|_1 + \|\tilde{s} - \hat{e}\|_1$. Since, the total variation distance computes the optimal distance of the spin fractions between two copies, this lower bounds the fraction of the Hamming distance. Therefore, we use the identity $\frac{1}{n} d_H(\sigma_t, \tilde{\sigma}_t) \geq \frac{1}{2}\|\alpha(\sigma_t) - \alpha(\tilde{\sigma}_t)\|_1$, (31), and (30) to deduce that for some constant $C_2 > 2q^2/\beta$

$$\Pr\left[J_i \neq \tilde{J}_i \mid \mathcal{F}_{i-1}\right] \leq \frac{\beta}{nq} d_H(\sigma_i, \tilde{\sigma}_i) + C_2\big(\|\alpha(\sigma_i) - \hat{e}\|_2^2 + \|\alpha(\tilde{\sigma}_i) - \hat{e}\|_2^2\big) + \frac{C_2}{n}.$$

Plugging the above equation in (29), we prove (28). Using Lemma 14 into (28), there exists a constant $C_3 > 0$ such that for any $t \geq n$,

$$\mathbb{E}\big[d_H(\sigma_t, \tilde{\sigma}_t)\big] \leq \frac{\beta}{nq} \sum_{i=t-n}^{t-1} \mathbb{E}[d_H(\sigma_i, \tilde{\sigma}_i)] + 2C_1C_3 \sum_{i=t-n}^{t-1} \gamma_n^{2i} + (2C_1C_3 + C_1).$$

Using the Lemma 17, we get that for some constant $C_4 > 0$, $\mathbb{E}\big[d_H(\sigma_t, \tilde{\sigma}_t)\big] = C_4 n \gamma_n^t + C_4$, and the result follows from Lemma 15 and Markov's inequality. ◀

References

1 Vedat Levi Alev and Ori Parzanchevski. Sequential sweeps and high dimensional expansion. *arXiv preprint*, 2023. doi:10.48550/arXiv.2312.02089.

2 Yali Amit. Convergence properties of the Gibbs sampler for perturbations of Gaussians. *The Annals of Statistics*, 24(1):122–140, 1996.

3 Antonio Blanca, Pietro Caputo, Alistair Sinclair, and Eric Vigoda. Spatial mixing and nonlocal Markov chains. *Random Structures & Algorithms*, 55(3):584–614, 2019. Extended abstract appeared SODA 2018. doi:10.1002/RSA.20844.

4 Antonio Blanca, Reza Gheissari, and Xusheng Zhang. Mean-field Potts and random-cluster dynamics from high-entropy initializations. *The Annals of Applied Probability*, 36(1):393–450, 2026. Extended abstract appeared SODA 2018.

5 Antonio Blanca and Md Tahmidur Rafid. Mixing and cutoff for the systematic scan dynamics of the mean-field ferromagnetic Potts model. *arXiv preprint*, 2026. arXiv:2607.09841.

6 Antonio Blanca and Zhezheng Song. Cutoff for the Swendsen–Wang Dynamics on the Complete Graph. In *45th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2025)*, pages 17:1–17:21, 2025. doi:10.4230/LIPIcs.FSTTCS.2025.17.

7 Antonio Blanca and Xusheng Zhang. Rapid Mixing of Global Markov Chains via Spectral Independence: The Unbounded Degree Case. *Algorithmica*, 88(1):19, 2026. Extended abstract appeared RANDOM 2023. doi:10.1007/S00453-025-01353-5.

8 J Bricmont. The statistical mechanics of lattice gases: Vol. 1. B. Simon, Princeton University Press, Princeton, New Jersey, 1993, 1996.

9 Iwona Chlebicka, Krzysztof Łatuszyński, and Błażej Miasojedow. Solidarity of Gibbs Samplers: the spectral gap. *The Annals of Applied Probability*, 35(1):142–157, 2025.

10 Paul Cuff, Jian Ding, Oren Louidor, Eyal Lubetzky, Yuval Peres, and Allan Sly. Glauber Dynamics for the Mean-Field Potts Model. *Journal of Statistical Physics*, 149(3):432–477, 2012.

11 Persi Diaconis and Arun Ram. Analysis of systematic scan Metropolis algorithms using Iwahori-Hecke algebra techniques. *Michigan Mathematical Journal*, 48(1):157–190, 2000.

12 Jian Ding, Eyal Lubetzky, and Yuval Peres. Censored Glauber dynamics for the mean field Ising model. *Journal of Statistical Physics*, 137(3):407–458, 2009.

13 Jian Ding, Eyal Lubetzky, and Yuval Peres. The mixing time evolution of Glauber dynamics for the mean-field Ising model. *Communications in Mathematical Physics*, 289(2):725–764, 2009.

14 Martin Dyer, Leslie Ann Goldberg, and Mark Jerrum. Systematic Scan for Sampling Colorings. *The Annals of Applied Probability*, pages 185–230, 2006.

15 Martin Dyer, Leslie Ann Goldberg, and Mark Jerrum. Dobrushin conditions and systematic scan. *Combinatorics, Probability and Computing*, 17(6):761–779, 2008. doi:10.1017/S0963548308009437.

16 Martin Dyer, Leslie Ann Goldberg, and Mark Jerrum. Matrix norms and rapid mixing for spin systems. *The Annals of Applied Probability*, 19(1):71–107, 2009.

17 Jason Gaitonde and Elchanan Mossel. Comparison theorems for the mixing times of systematic and random scan dynamics. In *Proceedings of the 37th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2575–2587. SIAM, 2026. doi:10.1137/1.9781611978971.93.

18 Andreas Galanis, Daniel Štefankovič, and Eric Vigoda. Swendsen-Wang algorithm on the mean-field Potts model. *Random Structures & Algorithms*, 54(1):82–147, 2019. Extended abstract appeared in RANDOM 2015. doi:10.1002/RSA.20768.

19 Heng Guo, Kaan Kara, and Ce Zhang. Layerwise systematic scan: Deep Boltzmann machines and beyond. In *International Conference on Artificial Intelligence and Statistics*, pages 178–187. PMLR, 2018. URL: http://proceedings.mlr.press/v84/guo18a.html.

20 Thomas P Hayes. A simple condition implying rapid mixing of single-site dynamics on spin systems. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 39–46. IEEE, 2006. doi:10.1109/FOCS.2006.6.

21 Bryan D He, Christopher M De Sa, Ioannis Mitliagkas, and Christopher Ré. Scan order in Gibbs sampling: Models in which it matters and bounds on how much. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, 2016.

22 Sanghak Jeon. Systematic scanning Glauber dynamics for the mean-field Ising model. *arXiv preprint*, 2023. arXiv:2307.10127.

23 David A Levin, Malwina J Luczak, and Yuval Peres. Glauber dynamics for the mean-field Ising model: cut-off, critical power law, and metastability. *Probability Theory and Related Fields*, 146(1):223, 2010.

24 David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, 2006.

25 Keunwoo Lim. Cutoff Phenomenon for Cyclic Dynamics on Hypercube. *arXiv preprint*, 2024. arXiv:2010.01756.

26 Yuval Peres and Peter Winkler. Can extra updates delay mixing? *Communications in Mathematical Physics*, 323(3):1007–1016, 2013.

27 Gareth O Roberts and Jeffrey S Rosenthal. Surprising Convergence Properties of Some Simple Gibbs Samplers under Various Scans. *International Journal of Statistics and Probability*, 5(4):112–121, 2016.

28 Sebastien Roch. *Modern Discrete Probability: An Essential Toolkit*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2024.

29 Ce Zhang and Christopher Ré. Towards high-throughput Gibbs sampling at scale: A study across storage managers. In *Proceedings of the 2013 ACM SIGMOD international conference on management of data*, pages 397–408, 2013. doi:10.1145/2463676.2463702.

Lifting Polynomial Complexity Measures Using Error-Correcting Codes

Ivan Hu
University of Wisconsin-Madison, WI, USA

Dieter van Melkebeek
University of Wisconsin-Madison, WI, USA

Abstract

We describe a method that lifts an arbitrary polynomial f with sparsity s to a polynomial that requires a read-once oblivious algebraic program of width s for every variable order. To do so, we introduce a technique for constructing a gadget based on erasure codes over finite fields, where each variable in f is substituted with a monomial that encodes a codeword into the exponents of the monomial. Our construction represents the first use of error-correcting codes in the context of lifting. As an application, we consider factor complexity, which studies how much the complexity of a polynomial can increase under factorization. Our result allows us to lift any gap in sparsity to the same gap in width in a generic manner.

2012 ACM Subject Classification Theory of computation → Algebraic complexity theory

Keywords and phrases lifting, lower bound, error-correcting code, read-once oblivious algebraic branching program, polynomial factorization

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.64

Category RANDOM

Funding This material is based upon work supported by the National Science Foundation under Grant Nos. 2137424 and 2312540.

Acknowledgements The authors thank Nutan Limaye for useful discussions, in particular for background on algebraic proof systems, and the anonymous reviewers for helpful suggestions.

1 Introduction

The concept of lifting involves taking a function that is hard for some computational model and transforming it in some fixed manner into a different function, such that this new function is guaranteed to be hard for a stronger computational model. Lifting has proven to be a useful concept in establishing lower bounds for several computational models [26, 15, 8], proof systems [9, 30, 7, 14, 20], as well as in communication complexity [16, 17, 24, 10]. Lifting is generally done with *gadgets*, which substitute each variable in the function with an expression in some new variables. In our setting, we consider lifting in terms of the complexity of polynomials in algebraic models of computation.

We consider two measures of complexity for a polynomial f. The first notion is *sparsity*, which is simply the number of monomials needed to write f as a linear combination of those monomials. The second notion is the minimum width needed to compute f with a *read-once oblivious algebraic branching program* (roABP), which is an algebraic analogue of read-once branching programs in the Boolean setting. The model is heavily studied in contexts such as polynomial identity testing [27, 13, 12, 1, 19, 18, 6] and algebraic lower bounds [25, 2, 23]. A structural feature of roABPs is that they read variables in a particular order, and the complexity of an roABP can vary depending on this variable order. Still, roABP width is always less than or equal to the sparsity no matter the variable order, making it a stronger complexity measure.


© Ivan Hu and Dieter van Melkebeek;
licensed under Creative Commons License CC-BY 4.0

Results. An efficient gadget for lifting a high sparsity polynomial to a polynomial requiring high roABP width has been constructed before [14, 20, 3], but only for certain variable orders for the roABP. An "intriguing question" raised in [3] is whether there is such a construction that produces a polynomial requiring high roABP width for *all* variable orders. In this work, we answer the question in the affirmative.

▶ **Theorem 1** (main result). *There exists an explicit gadget $\mathcal{G} : \mathbb{F}[x_1, \ldots, x_n] \to \mathbb{F}[y_1, \ldots, y_{2n}]$ such that for any polynomial $f(x_1, \ldots, x_n)$ of degree d and sparsity s, $\mathcal{G}(f)$ has degree $O(nd(n+d))$, sparsity s, and requires* roABP *width s in every variable order.*

Lifting results in the algebraic setting have consequences for closure properties, in particular for factor complexity: Given a polynomial f that is efficiently computable in some model, are all of the factors of f efficiently computable in the same model? A long line of work has established positive answers for reasonably strong models of algebraic computation, starting with general algebraic circuits [22], and later branching programs [29], formulas and constant-depth circuits [5], among others. However, in the weaker model of sparsity, there exists an explicit family of polynomials with a quasi-polynomial gap between the complexity of f and a factor of f [31] over any fixed field, and one with an exponential gap with specific conditions on the underlying fields [4]. The quasi-polynomial gap for sparsity was transformed into a quasi-polynomial gap for roABPs in [3] by exploiting the specific structure of the family. With our general lifting technique, any gap in factor complexity for sparsity generically translates into the same gap for roABPs.

Technique. The algebraic setting presents new opportunities for gadgets beyond the Boolean setting. For one thing, any Boolean function can be expressed as a multilinear polynomial over $\mathbb{F}_2$, but polynomials in general can have monomials with degrees that are arbitrarily large nonnegative integers. This allows us to encode information into the exponents of a monomial.

Our main idea is to use "codeword monomials" to replace each variable of the original polynomial f. More specifically, our gadget $\mathcal{G}$ substitutes each variable x_i with a monomial in new variables $y_1, \ldots, y_\ell$. A monomial $x_1^{d_1} \cdots x_n^{d_n}$ in f can be thought of as a vector $[d_1, \ldots, d_n] \in \mathbb{Z}^n$. The effect of the gadget can be interpreted as a linear map from $\mathbb{Z}^n \to \mathbb{Z}^\ell$ acting on each monomial of f. Our gadget essentially is a linear error-correcting code $E_p : \mathbb{F}_p^n \to \mathbb{F}_p^\ell$ for some sufficiently large prime p, where the elements of $\mathbb{F}_p$ are interpreted as elements of $\mathbb{Z}$ to serve as exponents.

How do we ensure that the sparsity of f becomes a lower bound for the roABP width of $\mathcal{G}(f)$ in every variable order? We use the well-known characterization of roABP width for a fixed variable order as the maximum rank of its coefficient matrices. Each coefficient matrix consists of the coefficients of the monomials of f up to the degree of f, arranged in a particular way. The injectivity of E guarantees that each monomial of f is mapped to a distinct monomial of $\mathcal{G}(f)$. By choosing E judiciously, we additionally ensure that for at least one of the coefficient matrices of $\mathcal{G}(f)$ in every order, the entries corresponding to those distinct monomials have distinct row indices and distinct column indices. Thus, if f has exactly s monomials, the s nonzero entries of this coefficient matrix are spread over distinct rows and distinct columns, which ensures that the matrix rank is s. As E_p is independent of the variable order, this yields a polynomial $\mathcal{G}(f)$ that requires roABP width at least s in every possible variable order.

We show that an erasure code E_p with appropriate parameters achieves our goal. The structure of the various coefficient matrices of $\mathcal{G}(f)$ depends on a partition of the variables $\{y_1, \ldots, y_\ell\}$ into two disjoint sets U and V. The row of the entry controlled by a given

monomial $\overline{m}$ is determined by the U-variables in $\overline{m}$, while the column is determined by the V-variables. With an erasure code gadget that can handle $|V|$ erasures, any $\overline{m}$ produced by $\mathcal{G}$ can be uniquely recovered from just knowing the U-variables. The effect is that no two nonzero entries share the same row. By swapping the roles of U and V, a similar property follows for columns instead of rows. An error-correcting code $E_p : \mathbb{F}_p^n \to \mathbb{F}_p^\ell$ that can handle erasures of up to half its output length ℓ has both properties for $|U| = |V| = \ell/2$. Reed-Solomon codes can handle the required number of erasures with $\ell = 2n$.

Organization. We start with the required background in Section 2, develop our results from first principles in Section 3, and end in Section 4 with directions for further research.

2 Preliminaries

We review elements of coding theory and the notions of polynomial sparsity and roABP's. As a warm-up, we also include a proof of the known gadget that works for restricted orders only. Throughout the paper, we use the notation $[n] = \{1, \ldots, n\}$ for positive integers n.

2.1 Codes

For a prime p, let $\mathbb{F}_p$ be the finite field of order p. For positive integers k, ℓ, a *linear error-correcting code* over $\mathbb{F}_p$ of dimension k and length ℓ is an injective linear map $E_p : \mathbb{F}_p^k \to \mathbb{F}_p^\ell$. We write $C \doteq E_p(\mathbb{F}_p^k)$ to denote the set of *codewords* of E_p. We say that E_p can *handle* e *erasures* if deleting up to e positions in a codeword of E_p still leaves enough information to retrieve the codeword. In symbols, for any two codewords $\mathbf{c}, \mathbf{c}' \in C$, and any subset of indices $I \subseteq [\ell]$ with $|I| \geq \ell - e$, we have that if $\mathbf{c}|_I = \mathbf{c}'|_I$, then $\mathbf{c} = \mathbf{c}'$.

We recall the properties of the classical Reed-Solomon code that we need. For completeness, we include the construction and proof.

▶ **Lemma 2** (Reed-Solomon, [28]). *For any positive integers k, e and prime $p \geq k + e$, there exists a linear code E_p over $\mathbb{F}_p$ with dimension k and length $k + e$ that can handle e erasures.*

Proof. Let $\alpha_1, \ldots, \alpha_{k+e}$ be fixed distinct elements in $\mathbb{F}_p$. For any vector $\mathbf{u} = \begin{bmatrix} u_1 & \ldots & u_k \end{bmatrix}$ in $\mathbb{F}_p^k$, define $f_{\mathbf{u}}(x) \in \mathbb{F}_p[x]$ to be the degree $k-1$ univariate polynomial

$$f_{\mathbf{u}}(x) = u_1 + u_2 x + \cdots + u_k x^{k-1}.$$

We define E as follows:

$$E_p(\mathbf{u}) = \begin{bmatrix} f_{\mathbf{u}}(\alpha_1) & \ldots & f_{\mathbf{u}}(\alpha_{k+e}) \end{bmatrix}.$$

We now show that E_p can handle e erasures. Let $I = \{i_1, \ldots, i_k\} \subseteq [k+e]$ be a set of $(k+e) - e = k$ indices. Suppose there are two codewords $\mathbf{c} = E_p(\mathbf{u})$, $\mathbf{c}' = E_p(\mathbf{u}')$ such that $\mathbf{c}|_I = \mathbf{c}'|_I$. Then for $j \in [k]$, we have that $f_{\mathbf{u}}(\alpha_{i_j}) = f_{\mathbf{u}'}(\alpha_{i_j})$. In other words, $f_{\mathbf{u}}$ and $f_{\mathbf{u}'}$ are degree $k-1$ polynomials that are equal at k distinct values. This implies that $f_{\mathbf{u}} = f_{\mathbf{u}'}$, meaning $\mathbf{c} = \mathbf{c}'$. ◀

Any element of $\mathbb{F}_p$ can be interpreted as an integer in $[0, p-1]$. For a linear code $E_p : \mathbb{F}_p^k \to \mathbb{F}_p^\ell$, we write the function $\widetilde{E}_p : \mathbb{F}_p^k \to \mathbb{Z}^\ell$ by interpreting the outputs of E_p as integers in this manner. Note that $\widetilde{E}_p$ is not a linear map, but if $E_p(\mathbf{u}) = \sum_{i=1}^s c_i E_p(\mathbf{v}_i)$, then $\widetilde{E}_p(\mathbf{u}) \equiv \sum_{i=1}^s c_i \widetilde{E}_p(\mathbf{v}_i) \bmod p$. Also, we abuse notation to allow vectors in $\mathbb{Z}^k$ as arguments to E_p and $\widetilde{E}_p$ by taking the entries modulo p.

2.2 Polynomials

Let $\mathbb{F}$ be a field (which may be unrelated to $\mathbb{F}_p$), and let $X = \{x_1, \ldots, x_n\}$ be a set of variables. A *monomial* $\overline{m}$ in X is a product of not necessarily distinct variables in X. For a subset $U \subseteq X$, define $\overline{m}_U$ to be the monomial produced by taking $\overline{m}$ and only including the variables in U (equivalently, setting $x_i = 1$ for $x_i \notin U$).

We can write a monomial uniquely as $\overline{m} = x_1^{d_1} \cdots x_n^{d_n}$, where $d_1, \ldots, d_n \geq 0$ are integers. We call the vector $[d_1, \ldots, d_n]$ the *degree vector* of $\overline{m}$. We refer to $\max_i(d_i)$ as the *individual degree* of $\overline{m}$, and to $\deg(\overline{m}) = \sum_i d_i$ as the *(total) degree* of $\overline{m}$. The individual and total degrees of a polynomial are defined as the maximum of the respective degrees of its monomials.

▶ **Definition 3** (sparsity). *For a polynomial $f \in \mathbb{F}[X]$, the sparsity of f is the number of terms used when f is written as a linear combination of monomials in X.*

2.3 Read-once oblivious algebraic branching programs

A *read-once oblivious algebraic branching program* (roABP) consists of a directed graph with $n+1$ layers labeled $0, 1, \ldots, n$. Layer 0 contains a source vertex s, layer n contains a sink vertex t, and each edge goes from a layer $i-1$ to layer i, for $i = 1, \ldots, n$. The other parameters are a set of n variables $x_1, \ldots, x_n$ and a variable order, which is a permutation $\pi : [n] \to [n]$. Each edge from layer $i-1$ to i is labeled with a univariate polynomial in $x_{\pi(i)}$. For every path γ from s to t, let the weight $\mathrm{wt}(\gamma)$ be the product of all the labels of the edges in γ. The polynomial f computed by the roABP is given by the sum of all the path weights from s to t:

$$f \doteq \sum_{\gamma : s \rightsquigarrow t} \mathrm{wt}(\gamma).$$

An roABP computes f only if both sides in the above equality are identical as polynomials in $\mathbb{F}[X]$, as opposed to just having the equality holding upon evaluation on all points in $\mathbb{F}^n$. If $|\mathbb{F}| > \deg(f)$, both notions are equivalent.

The *depth* of an roABP is the number of edge layers in the graph, i.e., n. The *width* of an roABP is equal to the maximum number of vertices in any layer. The *size* of an roABP is the total number of vertices in the graph. The primary measure of complexity of an roABP is width, since the depth is simply the number of variables, and the size is bounded by the product of width and depth.

▶ **Definition 4** (roABP width). *For a polynomial $f \in \mathbb{F}[X]$ and permutation $\pi : [n] \to [n]$, $\mathsf{roABP\text{-}width}_\pi(f)$ denotes the minimum width of any* roABP *of variable order π that computes f. $\mathsf{roABP\text{-}width}(f)$ denotes the minimum value of $\mathsf{roABP\text{-}width}_\pi(f)$ over all variable orders π.*

In our work, we do not directly use this definition to determine roABP width. Instead, we use a well-known characterization by Nisan [25] for non-commutative ABPs, extended to roABPs in [11, Lemma 4.5.8]. It expresses $\mathsf{roABP\text{-}width}_\pi(f)$ in terms of the ranks of its coefficient matrices.

▶ **Definition 5** (coefficient matrix). *Consider a partition $X \doteq U \sqcup V$. For a polynomial $f \in \mathbb{F}[X]$, we define a matrix $\mathrm{CMat}_{U,V}(f)$ as follows. The rows are indexed by monomials $\overline{m}_U$ over U, and the columns are indexed by monomials $\overline{m}_V$ over V. The entry of $\mathrm{CMat}_{U,V}(f)$ indexed by $(\overline{m}_U, \overline{m}_V)$ is the coefficient of $\overline{m}_U \overline{m}_V$ in f.*

Since we are only considering monomials of degree at most $\deg(f)$, we can assume that the dimensions of $\mathrm{CMat}_{U,V}(f)$ are finite.

▶ **Lemma 6** (roABP width characterization). *Let $f \in \mathbb{F}[X]$ be a polynomial, and let $\pi : [n] \to [n]$ be a variable order. For each $i \in \{0, \ldots, n\}$, let $U_i = \{x_{\pi(1)}, \ldots, x_{\pi(i)}\}$ and $V_i = \{x_{\pi(i+1)}, \ldots, x_{\pi(n)}\}$ be a partition of X. Then*

$$\mathsf{roABP\text{-}width}_\pi(f) = \max_{i \in \{0,\ldots,n\}} \operatorname{rank}(\mathrm{CMat}_{U_i,V_i}(f)).$$

roABP width is a stronger measure of complexity than sparsity. Specifically, any polynomial f with sparsity s satisfies $\mathsf{roABP\text{-}width}(f) \leq s$. To see this, one can construct an roABP with s disjoint paths or use Lemma 6 to see that any coefficient matrix has at most s nonzero entries, and thus has rank at most s.

The roABP width of a polynomial can vary wildly based on the variable order π. For example, consider the polynomial $f = \prod_{i=1}^{n}(x_i + y_i)$. If the variables are read in the order $x_1, y_1, \ldots, x_n, y_n$, then $\mathsf{roABP\text{-}width}_\pi(f) = 2$, while if the variables are read in the order $x_1, \ldots, x_n, y_1, \ldots, y_n$, then $\mathsf{roABP\text{-}width}_\pi(f) = 2^n$.

2.4 Lifting from sparsity to fixed-order roABP width

As a warm-up, we present a well-known gadget for lifting from sparsity to roABP width with respect to particular variable orders. Our gadget does not rely on it, but the construction was used in [14, 20, 3].

▶ **Proposition 7.** *There exists an explicit gadget $\mathcal{G} : \mathbb{F}[x_1, \ldots, x_n] \to \mathbb{F}[y_1, \ldots, y_n, z_1, \ldots, z_n]$ such that for any polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$ of total degree d and sparsity s, $\mathcal{G}_0(f)$ has the same individual degree at f, total degree $2d$, and $\mathsf{roABP\text{-}width}_\pi$ equal to s for every variable order π where the variables $y_1, \ldots, y_n$ are read before the variables $z_1, \ldots, z_n$.*

Proof. We define $\mathcal{G}(x_i) = y_i z_i$ for each $i \in [n]$, and write $g = \mathcal{G}(f)$. The individual degree of y_i and z_i in g are the same as the individual degree of x_i in f. The total degree of every monomial doubles, so $\deg(g) = 2d$.

For the roABP width, partition the variables $\{y_1, \ldots, y_n, z_1, \ldots, z_n\}$ into $Y = \{y_1, \ldots, y_n\}$, $Z = \{z_1, \ldots, z_n\}$. Notice that for any monomial $\overline{m}$ appearing in g, $\overline{m}_Y$ uniquely determines $\overline{m}_Z$ and vice versa, since $\overline{m}_Y$ and $\overline{m}_Z$ have the same degree vectors. Therefore, $\mathrm{CMat}_{Y,Z}(g)$ is a matrix with at most one nonzero entry in each row and column, and the number of nonzero entries is s. So $\operatorname{rank}(\mathrm{CMat}_{Y,Z}(g)) = s$ and Lemma 6 implies $\mathsf{roABP\text{-}width}_\pi(g) \geq s$. To see that $\mathsf{roABP\text{-}width}_\pi(g) \leq s$, notice that $\mathcal{G}$ maps monomials to monomials and thus g has sparsity at most s. ◀

3 Results

In this section we present our main result and its consequences for factor complexity.

3.1 Main result

Earlier gadgets for lifting from sparsity to roABP width either only work for restricted orders or require structural properties of the polynomials being lifted. We remedy these shortcomings and develop a gadget that works for every order and every polynomial.

Let $X = \{x_1, \ldots, x_n\}$, $Y = \{y_1, \ldots, y_{2n}\}$ be sets of variables. We define the gadget $\mathcal{G} : \mathbb{F}[X] \to \mathbb{F}[Y]$ for Theorem 1 as follows. For a prime $p > 2n$, let $E_p : \mathbb{F}_p^n \to \mathbb{F}_p^{2n}$ be a linear error-correcting code that can handle n erasures. Such a code exists by virtue of Lemma 2. Let $\widetilde{E}_p : \mathbb{F}_p^n \to \mathbb{Z}^{2n}$ be the corresponding integer-valued function. We write $\mathbf{e}_1, \ldots, \mathbf{e}_n \in \mathbb{F}_p^n$ as the standard basis vectors. We define $\mathcal{G}_p$ to perform the following substitution on a variable x_i.

$$\mathcal{G}_p(x_i) = \prod_{j=1}^{2n} y_j^{\widetilde{E}_p(\mathbf{e}_i)_j}. \tag{1}$$

Notice that if $\overline{m}$ is a monomial in X, then $\mathcal{G}_p(\overline{m})$ is a monomial in Y. The erasure capability of E allows us to prove the following structural lemma.

▶ **Lemma 8.** *Let n, d be positive integers, and $p > \max(2n, d)$ a prime. Let $Y \doteq U \sqcup V$ be any partition such that $|U| = |V| = n$. Let $\overline{m}_1, \overline{m}_2$ be monomials of individual degree at most d in X. If $\overline{m}_1 \neq \overline{m}_2$, then $\mathcal{G}_p(\overline{m}_1)_U \neq \mathcal{G}_p(\overline{m}_2)_U$ and $\mathcal{G}_p(\overline{m}_1)_V \neq \mathcal{G}_p(\overline{m}_2)_V$.*

Proof. The condition that $p > 2n$ ensures that the gadget $\mathcal{G}_p$ is well-defined.

Let $\overline{m}$ be a monomial in X with degree vector $\mathbf{d} = [d_1, \ldots, d_n]$. Since $d_1 E_p(\mathbf{e}_1) + \cdots + d_n E_p(\mathbf{e}_n) = E_p(\mathbf{d})$, we have that $d_1 \widetilde{E}_p(\mathbf{e}_1) + \cdots + d_n \widetilde{E}_p(\mathbf{e}_n) \equiv \widetilde{E}_p(\mathbf{d}) \bmod p$. In other words, there exist integers $q_1, \ldots, q_{2n}$ such that

$$\mathcal{G}_p(\overline{m}) = \prod_{j=1}^{2n} y_j^{d_1 \widetilde{E}_p(\mathbf{e}_1)_j + \cdots + d_n \widetilde{E}_p(\mathbf{e}_n)_j} = \prod_{j=1}^{2n} y_j^{\widetilde{E}_p(\mathbf{d})_j + pq_j}.$$

Armed with this fact, let $\overline{m}_1, \overline{m}_2$ be monomials of degree at most d in X. Suppose $\overline{m}_1, \overline{m}_2$ have degree vectors $\mathbf{d}_1$, $\mathbf{d}_2$, respectively. Furthermore, suppose that for $U = \{y_{i_1}, \ldots, y_{i_n}\} \subseteq Y$, we have that $\mathcal{G}_p(\overline{m}_1)_U = \mathcal{G}_p(\overline{m}_2)_U$. In other words, the powers of the U-variables in $\mathcal{G}_p(\overline{m}_1)$ and $\mathcal{G}_p(\overline{m}_2)$ agree. This means that, for $j \in [n]$, $\widetilde{E}_p(\mathbf{d}_1)_{i_j} \equiv \widetilde{E}_p(\mathbf{d}_2)_{i_j} \bmod p$, or $E_p(\mathbf{d}_1)_{i_j} = E_p(\mathbf{d}_2)_{i_j}$. So, $E_p(\mathbf{d}_1)$ and $E_p(\mathbf{d}_2)$ agree on at least n indices. Because E_p has output length $2n$ and can handle n erasures, we have that $\mathbf{d}_1 \equiv \mathbf{d}_2 \bmod p$. Since $p > d$ and each entry of $\mathbf{d}_1, \mathbf{d}_2$ is at most d, we have that $\mathbf{d}_1 = \mathbf{d}_2$. Therefore, if $\mathcal{G}_p(\overline{m}_1)_U = \mathcal{G}_p(\overline{m}_2)_U$, then $\overline{m}_1 = \overline{m}_2$.

We can conclude that if $\overline{m}_1 \neq \overline{m}_2$, then $\mathcal{G}_p(\overline{m}_1)_U \neq \mathcal{G}_p(\overline{m}_2)_U$. By the same reasoning, we can see that $\mathcal{G}_p(\overline{m}_1)_V \neq \mathcal{G}_p(\overline{m}_2)_V$. ◀

Now we show that our gadget $\mathcal{G}_p$ satisfies the conditions of Theorem 1. In fact, we prove a refined statement with Theorem 9 that considers the individual degree of each variable.

▶ **Theorem 9** (refined main result). *There exists an explicit gadget $\mathcal{G} : \mathbb{F}[x_1, \ldots, x_n] \to \mathbb{F}[y_1, \ldots, y_{2n}]$ such that for any polynomial $f(x_1, \ldots, x_n)$ of degree d and sparsity s, $\mathcal{G}(f)$ has individual degree $O(d(n+d))$ in each variable, sparsity s, and* roABP-width$(\mathcal{G}(f)) = s$.

Proof. Let $m \doteq \max(2n, d) + 1$. By Bertrand's postulate, there always exists a prime $p \in [m, 2m]$. We show that $\mathcal{G} \doteq \mathcal{G}_p$ satisfies the conditions of the theorem.

Write $g = \mathcal{G}(f)$. First, we prove the degree bound. Since $\widetilde{E}_p(\mathbf{e}_i)_j < p \leq 2m$ for any $i \in [n]$, $j \in [2n]$, $\mathcal{G}(x_i)$ is a monomial in Y of individual degree less than $2m$. As each monomial in f contains at most d occurrences of variables (counting multiplicities), after replacing each x_i by $\mathcal{G}(x_i)$, the resulting monomials have individual degree less than $md = O(d(n+d))$.

Now we show the lower bound on roABP width. Let $\pi : [2n] \to [2n]$ be any variable order on Y. Let $U = \{y_{\pi(1)}, \ldots, y_{\pi(n)}\}$ and $V = \{y_{\pi(n+1)}, \ldots, y_{\pi(2n)}\}$. We reason about the structure of the coefficient matrix $\mathrm{CMat}_{U,V}(g)$. Since f has sparsity s, we can write it as a linear combination of s monomials: $f = \sum_{i=1}^{s} c_i \overline{m}_i$ for some $c_i \neq 0$. We have

$$\mathcal{G}(f) = \sum_{i=1}^{s} c_i \mathcal{G}(\overline{m}_i) = \sum_{i=1}^{s} c_i \mathcal{G}(\overline{m}_i)_U \mathcal{G}(\overline{m}_i)_V.$$

By Lemma 8, all monomials in the sets $\{\mathcal{G}(\overline{m}_1)_U,\ldots,\mathcal{G}(\overline{m}_s)_U\}$ and $\{\mathcal{G}(\overline{m}_1)_V,\ldots,\mathcal{G}(\overline{m}_s)_V\}$ are distinct. Thus, for $i \in [s]$, the entry $(\mathcal{G}(\overline{m}_i)_U, \mathcal{G}(\overline{m}_i)_V)$ of $\mathrm{CMat}_{U,V}(g)$ is nonzero, all other entries of $\mathrm{CMat}_{U,V}(g)$ are zero, and no two of the s nonzero entries share a row or column. This implies that $\mathrm{rank}(\mathrm{CMat}_{U,V}(g)) \geq s$, and by Lemma 6, this implies that $\mathsf{roABP\text{-}width}_\pi(g) \geq s$. Since π was arbitrary, we have that $\mathsf{roABP\text{-}width}(g) \geq s$.

Finally, since $\mathcal{G}$ maps monomials to monomials, g still has sparsity s, so there exists an roABP of width s computing g, no matter the variable order. Therefore, $\mathsf{roABP\text{-}width}(g) = s$. ◀

3.2 Application to factor complexity

We now establish the consequences for factor complexity that we mentioned in the introduction. They immediately follow from our main result and two well-known gaps for sparsity.

Over any field, the following gap in sparsity was established in [31].

▶ **Fact 10.** For all fields $\mathbb{F}$ and positive integers n and d, there exists a degree-nd polynomial $f(x_1,\ldots,x_n)$ of sparsity 2^n that has a factor $g(x_1,\ldots,x_n)$ of sparsity d^n. Specifically,

$$f = \prod_{i=1}^{n}(x_i^d - 1) \text{ and } g = \prod_{i=1}^{n}(1 + x_i + x_i^2 + \cdots + x_i^{d-1}).$$

Over a field of nonzero characteristic d, the identity $(x_1 + \cdots + x_n)^d = x_1^d + \cdots + x_n^d$ yields the following gap (see, e.g., [4, Example 4.7]).

▶ **Fact 11.** Suppose $d = \mathrm{char}(\mathbb{F}) \neq 0$. For all positive integers n, there exists a degree-d polynomial $f(x_1,\ldots,x_n)$ of sparsity n that has a factor $g(x_1,\ldots,x_n)$ of sparsity $\binom{n+d-2}{d-1}$. Specifically,

$$f = x_1^d + \cdots + x_n^d \text{ and } g = (x_1 + \cdots + x_n)^{d-1}.$$

Notice that by the multinomial theorem and the fact that d is prime, every monomial of degree up to $d-1$ occurs in g with a nonzero coefficient modulo d.

By lifting Fact 10 and Fact 11 using our main result, we obtain the following results for roABP width.

▶ **Corollary 12.** *Let $\mathbb{F}$ be a field and n a positive integer.*

(a) *For every positive integer d, there exists a degree-$O(d^2n^3)$ polynomial $F(y_1,\ldots,y_{2n})$ such that $\mathsf{roABP\text{-}width}(F)=2^n$, and F has a factor $G(y_1,\ldots,y_{2n})$ such that $\mathsf{roABP\text{-}width}(G) = d^n$.*

(b) *If $\mathrm{char}(\mathbb{F}) = d \neq 0$, then there exists a degree-$O(nd(n+d))$ polynomial $F(y_1,\ldots,y_{2n})$ such that $\mathsf{roABP\text{-}width}(F) = n$, and F has a factor $G(y_1,\ldots,y_{2n})$ such that $\mathsf{roABP\text{-}width}(G) = \binom{n+d-2}{d-1}$.*

Proof. Since $\mathcal{G}$ is a substitution, if g is a factor of f, then $G \doteq \mathcal{G}(g)$ is a factor of $F \doteq \mathcal{G}(f)$. Apply Theorem 1 to Fact 10 and Fact 11. Respectively, we obtain $\mathsf{roABP\text{-}width}(F) = 2^n$ and $\mathsf{roABP\text{-}width}(G) = d^n$, and $w \doteq \mathsf{roABP\text{-}width}(F) = n$ and $\mathsf{roABP\text{-}width}(G) = \binom{n+d-2}{d-1}$. ◀

Note that for fixed d, the width gap in part (a) of Corollary 12 goes from $w = 2^n$ to $d^n = w^{\Omega(\log d)}$, and in part (b) from $w = n$ to $\binom{n+d-2}{d-1} = w^{\Omega(d)}$. By instead setting $d = n$, we obtain an explicit family of polynomials with a quasi-polynomial gap $w \mapsto w^{\Omega(\log w)}$ in case (a), and an exponential gap $w \mapsto \binom{2w-2}{w-1} = \Theta(2^{2w}/\sqrt{w})$ in case (b).

In comparison to the approach in [3], our technique has the advantage of exactly preserving the complexity measure, while [3] weakens the complexity of the factor slightly. Furthermore, we can apply our methods to any family of polynomials, while [3] requires specific conditions on the family. On the other hand, [3] has the advantage that the degree of the polynomial increases by a constant factor, while our gadget increases the degree by a factor of $O(n(n+d))$.

4 Further research

Error-correcting codes have been helpful in many contexts. We exhibit their first use in the context of lifting and manage to solve an open problem in lifting raised by [3]. Can our technique be applied to other lifting results, as well? For example, does it yield generic lifting lower bounds for roABPs in the context of algebraic proof systems such as fragments of the Ideal Proof System (IPS)? Previous works in this area [14, 20, 21] established lower bounds for specific instances of roABP-IPS. An interesting open question is to construct a gadget that takes a generic hard instance for a weaker fragment of IPS and lifts it to an instance that is hard for roABP-IPS.

On a more technical level, in comparison to other examples of lifting like [3], the gadget in our work increases the degree of the polynomial by a relatively large factor, polynomial in the original degree d and number of variables n. Could one construct a more efficient gadget that lifts sparsity to roABP width? For example, can one construct a gadget that only uses multilinear codeword monomials?

In the context of factor gap complexity, for degree-d polynomials of sparsity s, the best known gaps are essentially the ones given in Section 3.2, namely that a factor can have sparsity $s^{\Omega(\log d)}$ over any field and $s^{\Omega(d)}$ over fields with nonzero characteristic d. [4] showed an upper bound of $s^{O(d^2 \log n)}$ for the sparsity of any factor over any field, where n is the number of variables. Can one construct examples that achieve larger gaps in sparsity and therefore, by our main result, in roABPs width? Alternatively, can one find larger gaps in roABPs width independent of any results on sparsity?

References

1 Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *SIAM J. Comput.*, 44(3):669–697, 2015. doi:10.1137/140975103.

2 Matthew Anderson, Michael A. Forbes, Ramprasad Saptharishi, Amir Shpilka, and Ben Lee Volk. Identity testing and lower bounds for read-k oblivious algebraic branching programs. *ACM Trans. Comput. Theory*, 10(1):3:1–3:30, 2018. doi:10.1145/3170709.

3 Robert Andrews, Jules Armand, Prateek Dwivedi, Magnus Rahbek Dalgaard Hansen, Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. On closure properties of read-once oblivious algebraic branching programs. In *17th Innovations in Theoretical Computer Science Conference, ITCS 2026, Bocconi University, Milan, Italy, January 27-30, 2026*, LIPIcs, pages 9:1–9:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. doi:10.4230/LIPIcs.ITCS.2026.9.

4 Vishwas Bhargava, Shubhangi Saraf, and Ilya Volkovich. Deterministic factorization of sparse polynomials with bounded individual degree. *J. ACM*, 67(2):8:1–8:28, 2020. doi:10.1145/3365667.

5 Somnath Bhattacharjee, Mrinal Kumar, Shanthanu S. Rai, Varun Ramanathan, Ramprasad Saptharishi, and Shubhangi Saraf. Closure under factorization from a result of Furstenberg. *CoRR*, abs/2506.23214, 2025. doi:10.48550/arXiv.2506.23214.

6 Pranav Bisht and Nitin Saxena. Blackbox identity testing for sum of special ROABPs and its border class. *Comput. Complex.*, 30(1):8, 2021. doi:10.1007/S00037-021-00209-Y.

7 Yogesh Dahiya, Meena Mahajan, and Sasank Mouli. New lower bounds for polynomial calculus over non-Boolean bases. In *27th International Conference on Theory and Applications of Satisfiability Testing, SAT 2024, Pune, India, August 21-24, 2024*, LIPIcs, pages 10:1–10:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.SAT.2024.10.

8 Susanna F. de Rezende, Mika Göös, and Robert Robere. Guest column: Proofs, circuits, and communication. *SIGACT News*, 53(1):59–82, 2022. doi:10.1145/3532737.3532746.

9 Susanna F. de Rezende, Or Meir, Jakob Nordström, Toniann Pitassi, Robert Robere, and Marc Vinyals. Lifting with simple gadgets and applications to circuit and proof complexity. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 24–30. IEEE, 2020. doi:10.1109/FOCS46700.2020.00011.

10 Susanna F. de Rezende and Marc Vinyals. Lifting with colourful sunflowers. In *40th Computational Complexity Conference, CCC 2025, Toronto, Canada, August 5-8, 2025*, LIPIcs, pages 36:1–36:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.CCC.2025.36.

11 Michael A. Forbes. *Polynomial identity testing of read-once oblivious algebraic branching programs*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2014. URL: https://hdl.handle.net/1721.1/89843.

12 Michael A. Forbes, Ramprasad Saptharishi, and Amir Shpilka. Hitting sets for multilinear read-once algebraic branching programs, in any order. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 867–875. ACM, 2014. doi:10.1145/2591796.2591816.

13 Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, Berkeley, CA, USA, October, 26-29, 2013*, pages 243–252. IEEE Computer Society, 2013. doi:10.1109/FOCS.2013.34.

14 Michael A. Forbes, Amir Shpilka, Iddo Tzameret, and Avi Wigderson. Proof Complexity Lower Bounds from Algebraic Circuit Complexity. *Theory of Computing*, 17:1–88, 2021. doi:10.4086/TOC.2021.V017A010.

15 Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from resolution. *Theory Comput.*, 16:1–30, 2020. doi:10.4086/TOC.2020.V016A013.

16 Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. *SIAM J. Comput.*, 47(6):2435–2450, 2018. doi:10.1137/16M1059369.

17 Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for BPP. *SIAM J. Comput.*, 49(4), 2020. doi:10.1137/17M115339X.

18 Zeyu Guo and Rohit Gurjar. Improved explicit hitting-sets for roabps. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, Virtual Conference, August 17-19, 2020*, LIPIcs, pages 4:1–4:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.4.

19 Rohit Gurjar, Arpita Korwar, Nitin Saxena, and Thomas Thierauf. Deterministic identity testing for sum of read-once oblivious arithmetic branching programs. *Comput. Complex.*, 26(4):835–880, 2017. doi:10.1007/S00037-016-0141-Z.

20 Tuomas Hakoniemi, Nutan Limaye, and Iddo Tzameret. Functional lower bounds in algebraic proofs: Symmetry, lifting, and barriers. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 1396–1404. ACM, 2024. doi:10.1145/3618260.3649616.

21 Tuomas Hakoniemi, Nutan Limaye, and Iddo Tzameret. Hard CNF instances for ideal proof systems. *CoRR*, abs/2605.04544, 2026. doi:10.48550/arXiv.2605.04544.

22 Erich L. Kaltofen. Factorization of polynomials given by straight-line programs. *Adv. Comput. Res.*, 5:375–412, 1989.

23 Neeraj Kayal, Vineet Nair, and Chandan Saha. Separation between read-once oblivious algebraic branching programs (ROABPs) and multilinear depth-three circuits. *ACM Trans. Comput. Theory*, 12(1):2:1–2:27, 2020. doi:10.1145/3369928.

24 Shachar Lovett, Raghu Meka, Ian Mertz, Toniann Pitassi, and Jiapeng Zhang. Lifting with sunflowers. In *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, Berkeley, CA, USA, January 31 - February 3, 2022*, LIPIcs, pages 104:1–104:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ITCS.2022.104.

25 Noam Nisan. Lower bounds for non-commutative computation. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 410–418. ACM, 1991. doi:10.1145/103418.103462.

26 Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Comb.*, 19(3):403–435, 1999. doi:10.1007/S004930050062.

27 Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Comput. Complex.*, 14(1):1–19, 2005. doi:10.1007/S00037-005-0188-8.

28 Irving S. Reed and Gustave Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960. doi:10.1137/0108018.

29 Amit Sinhababu and Thomas Thierauf. Factorization of polynomials given by arithmetic branching programs. *Computational Complexity*, 30(2):15:1–15:47, 2021. doi:10.1007/S00037-021-00215-0.

30 Dmitry Sokolov. (Semi)Algebraic proofs over $\{\pm 1\}$ variables. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 78–90. Association for Computing Machinery, 2020. doi:10.1145/3357713.3384288.

31 Joachim von zur Gathen and Erich L. Kaltofen. Factoring sparse multivariate polynomials. *J. Comput. Syst. Sci.*, 31(2):265–287, 1985. doi:10.1016/0022-0000(85)90044-3.

Property Testing of Computational Networks

Artur Czumaj
Department of Computer Science and DIMAP, University of Warwick, Coventry, UK
University of Cologne, Germany

Christian Sohler
University of Cologne, Germany

Abstract

In this paper we initiate the study of *property testing of weighted computational networks viewed as computational devices*. Our goal is to design property testing algorithms that for a given computational network with oracle access to the weights of the network, accept (with probability at least $\frac{2}{3}$) any network that computes a certain function (or a function with a certain property) and reject (with probability at least $\frac{2}{3}$) any network that is *far* from computing the function (or any function with the given property). We parameterize the notion of being far and want to reject networks that are (ε, δ)-*far*, which means that one needs to change an ε-fraction of the description of the network to obtain a network that computes a function that differs in at most a δ-fraction of inputs from the desired function (or any function with a given property).

To exemplify our framework, we present a case study involving simple neural Boolean networks with ReLU activation function. As a highlight, we demonstrate that for such networks, any near-constant function is testable in query complexity independent of the network's size. We also show that a similar result cannot be achieved in a natural generalization of the distribution-free model to our setting, and also in a related vanilla testing model. While the analysis of neural Boolean networks with ReLU activation function is rigorous and technically demanding, we consider it a testament to the robustness of our framework and the depth of our findings.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms; Computing methodologies → Neural networks

Keywords and phrases Property Testing, Testing Computational Networks

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.65

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2512.07577` [19]

Funding *Artur Czumaj*: Research partially supported by the Key Profile Area (KPA): "Intelligent Methods for Earth System Sciences" at the University of Cologne and by the Centre for Discrete Mathematics and its Applications (DIMAP) at the University of Warwick.

Acknowledgements We would like to thank Gereon Frahling for his valuable discussions regarding the computational model. We would like to thank Nithin Varma for helpful discussions during the initial phase of research that lead to this paper.

1 Introduction

Since its development in the late 90s [46, 29], the area of *property testing* has emerged as an important area of modern theoretical computer science. It considers the relaxation of classical decision problems by distinguishing inputs that belong to a given set (have a given property $\mathcal{P}$) from those that are *"far"* from any input that belongs to the set (are *far* from having property $\mathcal{P}$). In its most classical setting (see, e.g., survey expositions in [28, 9]), a property $\mathcal{P}$ is a collection of objects (binary strings, graphs, etc), and being *"far"* is measured by the Hamming distance, namely, in how many places of its representation should an input object


© Artur Czumaj and Christian Sohler;
licensed under Creative Commons License CC-BY 4.0

be changed so as to have the property $\mathcal{P}$. As a useful and powerful way of relaxation of the classical decision problems, property testing has found numerous applications and established itself as a significant topic in the study of function properties, graph properties, and beyond. Examples include testing properties of functions such as being a low degree polynomial, being monotone, depending on a specified number of attributes, testing properties of graphs such as being bipartite and being triangle-free, testing properties of strings, and testing properties of geometric objects and visual images such as being well-clustered and being convex (for a more complete picture, see, e.g., [28, 9, 16, 18, 45] and the references therein). Furthermore, property testing is in the center of recent advances in *sublinear algorithms*, as property testing algorithms are frequently "superfast" and rely on inspecting small portions of objects and making the evaluation based on such an inspection.

In property testing one assumes an indirect access to the input and its representation that is provided by an *oracle*; e.g., one can query the value of a function on a *specific input*, or the existence of a *specific edge* of a graph. The *complexity of the tester* is the number of queries to the oracle.

In this paper, *we study property testing in the context of complex computational devices* (e.g., *computational networks*). We consider a setting in which one wants to study functions that are represented in an indirect way (e.g., as circuits or computational networks, or via complicated algebraic expressions) by computational devices used to determine the value of the functions. One may think about functions that are given *implicitly*, and in order to compute f one has to perform a possibly non-trivial computation on the input. For example, we may have a computational network that *computes* a function $f : \mathbb{R}^n \to \{0,1\}$; or we may have a function $f : \{0,1\}^n \to \mathbb{R}^r$ defined by a matrix $A \in \mathbb{R}^{r\times n}$ as $f(x) := Ax$, or $f : \{0,1\}^n \to \{0,1\}^r$ defined as[1] $f(x) := \mathrm{sgn}(\mathrm{ReLU}(Ax))$.

Our goal is to determine whether a given computational device computes a function having a given property or if it is *far* from any computational device computing function with a given property. For example, if for a matrix $A \in \mathbb{R}^{r\times n}$ we have $f(x) := Ax$, then, informally, we would call the "device" *Ax far from having property $\mathcal{P}$* if there is no matrix $A^* \in \mathbb{R}^{r\times n}$ that differs from A only in a small fraction of its entries (i.e., $\|A^* - A\|_0$ is small) such that function $f^*(x) := A^*x$ has the property $\mathcal{P}$. Furthermore, to make our study more general and applicable, we will allow to enhance our definition to devices that err on a small number of inputs.

Observe that while our model can be seen as a special case of the classical property testing framework (see, e.g., references to a similar setting in [28, Chapter 1]), the approach presented in our paper offers a different perspective than that most frequently studied: *our framework focuses on the computational device* – and its analysis should depend on specific parameters of the device rather than on the inputs or the function f. We want to determine if one has to make many changes in the device's representations in order to compute some function having desirable property $\mathcal{P}$, or only a small number of changes suffices. And so, for example, if for a matrix $A \in \mathbb{R}^{r\times n}$ we define $f(x) := Ax$ or $f(x) := \mathrm{sgn}(\mathrm{ReLU}(Ax))$ for any $x \in \mathbb{R}^n$, then it is natural to define the oracle to access individual entries from A (rather than to allow queries to the value of $f(x)$ for any specific input x).

[1] We use standard **ReLU** and **signum** functions to ensure that f has range $\{0,1\}^r$; recall that $\mathrm{ReLU}(z) := \max\{z, 0\}$ and $\mathrm{sgn}(z) := 1$ if $z > 0$, $\mathrm{sgn}(0) := 0$, $\mathrm{sgn}(z) := -1$ if $z < 0$. For vectors, the operations are for each coordinate. (Note that the use of these standard definitions make the combination $\mathrm{sgn}(\mathrm{ReLU}(\cdot))$ *not* redundant: $\mathrm{sgn}(\mathrm{ReLU}(Ax)) \not\equiv \mathrm{sgn}(Ax)$ for matrices A containing *negative* values – as it is the case for matrices and vectors studied in our paper.)

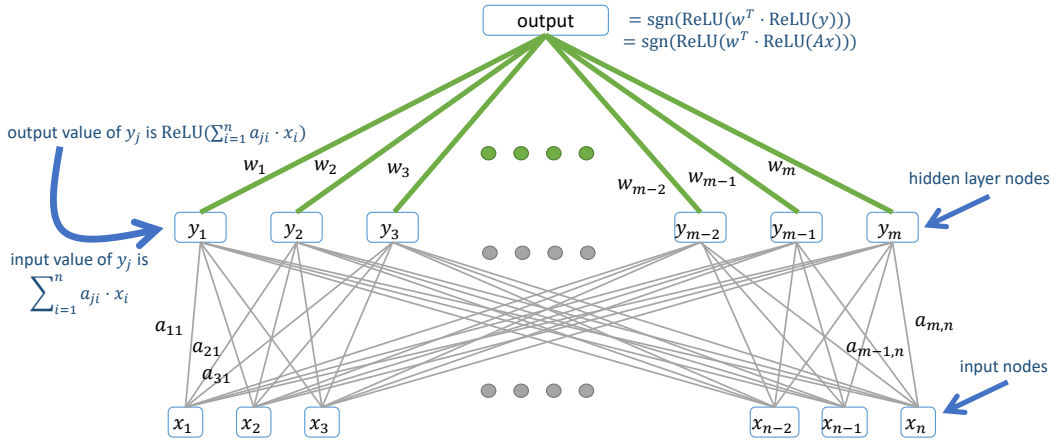


Figure 1 A ReLU network with n input nodes with inputs $x_1, \ldots, x_n$, a single hidden layer with m hidden layer nodes with values $y_1, \ldots, y_m$, and one output node. Every input node $x_i \in \{0,1\}$ is connected to every hidden layer node y_j by an edge with real weight $a_{ji} \in [-1,1]$ and every hidden layer node y_j is connected to the output node by an edge of real weight $w_j \in [-1,1]$. The value of node y_j is $\sum_{i=1}^n a_{ji}x_i$, which after applying the ReLU activation function gives $\mathrm{ReLU}(\sum_{i=1}^n a_{ji}x_i)$. The Boolean function computed by the network is equal to $\mathrm{sgn}\left(\mathrm{ReLU}\left(\sum_{j=1}^m \left(w_j \cdot \mathrm{ReLU}(\sum_{i=1}^n a_{ji}x_i)\right)\right)\right) \equiv \mathrm{sgn}\left(\mathrm{ReLU}(w^T \cdot \mathrm{ReLU}(Ax))\right)$. A more general ReLU network with multiple inputs, multiple outputs, and multiple layers is depicted on Figure 2.

Case study: Testing basic neural ReLU networks. Our setting is very open-ended, and in order to make our study more concrete, we will consider our framework for a specific example of complex computational networks. Our goal is to take a representative example of computational networks that is generic and simple, yet complex enough to go beyond the study relying on random sampling. We will study an example of the most basic yet highly non-trivial *neural ReLU networks* seen as computational devices from *the point of view of property testing*. Consider a trained feedforward network and view it as a circuit that receives an input vector $x \in \{0,1\}^n$ and computes some output. In the most basic case of the network with one output, one hidden layer, and the ReLU activation function (see also Figure 1), when a network is interpreted as a binary classifier, for a vector $w \in [-1,1]^m$ and matrix $A \in [-1,1]^{m \times n}$ it computes a Boolean function of the form

$$f(x) := \mathrm{sgn}\left(\mathrm{ReLU}(w^T \cdot \mathrm{ReLU}(Ax))\right).$$

(Our setting can be naturally extended to the study of more complex neural networks (functions) with *multiple outputs* and *multiple hidden layers*, where for matrices $W_0, \ldots, W_\ell$ with $W_i \in [-1,1]^{m_{i+1} \times m_i}$, we compute a function $f : \{0,1\}^{m_0} \to \{0,1\}^{m_{\ell+1}}$ defined as $f(x) := \mathrm{sgn}(\mathrm{ReLU}(W_\ell \cdot \ \ldots \ \cdot \mathrm{ReLU}(W_1 \cdot \mathrm{ReLU}(W_0 \cdot \mathrm{ReLU}(x))) \ldots))$, see Section 3.4.)

In order to facilitate our analysis in the property testing framework, we will consider property testing algorithms that are given the oracle access to the weights of the network[2] and the objective is to accept networks that compute functions that have a certain property of interest and reject networks that compute functions that are far from having that property for almost all inputs.

[2] Which for networks with one output and one hidden layer, is to query for the entries of A and w.

Our motivation. Our study is motivated by the modeling of computational networks in the framework of property testing, and to instantiate our framework in the context of basic neural networks. Taking the point of view of property testing and complexity theory, our goal is to obtain a better understanding of the relation between the (local) structure of a computational network and the function it computes. Our motivation stems from the fact that property testing has been a successful tool to reveal relations between the local structure of an object (here, a computational device) and its global properties that can provide combinatorial insights about the structure of an object. Examples of such insights include the characterization of property testing for dense graphs and its relation to the regularity lemma [1, 3], the relation of property testing of hyperfinite graphs to the theory of amenable groups [23, 39], or the line of research that started with testing connectivity [30] and then studied sublinear algorithms for approximating MST cost [13, 15, 17] and whose structural findings formed a building block in finding the currently fastest approximation algorithms for low dimensional MST approximation [4].

To showcase the versatility and power of our framework, we study *networks that are simple*, yet whose *computational properties are complex*. The reliance on *non-linear ReLU operators* is not only something that makes neural deep networks so powerful, but this also makes their study challenging (observe that, say, for $f(x) := Ax$, simple random sampling of entries of matrix A would provide sufficient information to understand many properties of such function; the non-linear ReLU operators make the study and the analysis more complex and interesting). Feedforward neural networks, as those studied in the simplest form in our paper, form a fundamental part of modern deep networks [34] such as transformers and the attention mechanism [49], which in turn are the main building block of large language models such as ChatGPT [43]. Unfortunately, despite their relatively simple setting, we still lack a good understanding of the computation process of these networks, leading to serious questions on the trust and transparency of the underlying learning algorithms. Therefore, even though *our main focus is on the complexity study*, we hope that the lens of property testing will contribute to a better understanding of the relation between the network structure and the computed function or properties of that function. Since the analysis of property testing algorithms relies on establishing close links between the object's local structure and the tested property, we believe that the study of simple yet fundamental models of ReLU networks may provide a new insight into such networks. Additionally, the basic notion of a relaxed decision problem in property testing has some appealing properties in the context of understanding neural networks: Typically, we would like a neural network to be robust to small changes in the network, as otherwise there is a higher risk of adversarial attacks [48]. Furthermore, during network training, *dropout* [47] is often used to improve generalization, i.e., a random fraction of input nodes is dropped during training for each training example.

1.1 Overview of our results

In this paper, we *introduce a property testing model for computational networks* through a study of *basic neural networks*. We first focus on a simple network structure: a fully connected feedforward network with the ReLU activation function, one hidden layer, and one output node (see Figure 1 and Definition 1). Later, we generalize our model to a more complex setting with multiple hidden layers and multiple output nodes (see Section 2.1 and Figure 2). We view the network as an *arithmetic circuit* that computes a Boolean function. Our objective is to design property testing algorithms that sample weights of the network and accept, if the network computes a function with a specific property, and reject, if the network is (ε, δ)-far from computing any function with the property, by which we mean that one has to change more than an ε-fraction of the network to obtain a function that differs from the target functions on less than a δ-fraction of the inputs.

An important contribution of this paper is *introduction of a model that allows a non-trivial study of computational networks (e.g., neural networks) from the lens of property testing.*

Next, we provide a *comprehensive study of our model*, focusing on the example of the constant 0 and the OR-function, for which we design constant-complexity property testers. Further, we consider other natural variants of our model for which we provide super-constant lower bounds demonstrating their complexity limitations. This shows that our model introduces a useful notion of being far that leads to interesting questions and results from the algorithmic and complexity perspective. Our results extend beyond the 0/OR-functions and we give some positive results showing the testability of some Boolean functions and classes of functions.

Our *main technical contributions* are (see Section 3 for more details):

(1) study of our model on the problem of *testing some very simple yet fundamental functions*,
(2) study of limitations of possibility of extending our results to a *distribution-free model*,
(3) *first steps towards understanding of constant time testable properties* in our model,
(4) extending the model to *more complex networks with multiple outputs and layers.*

In the following, after reviewing related work, we introduce our property testing model in detail. Then, in Section 3, we summarize our results and provide a technical overview. Due to space constraints, we omit the full technical details for the claims made in Section 3. However, to illustrate our analytical approach, Section 4 provides a complete proof of one key result. All remaining proofs and a comprehensive analysis are deferred to the full version [19].

1.2 Related work

Property testing. Since its development in the late 90s [46, 29], the area of property testing has emerged as an important area of modern theoretical computer science, see surveys in [9, 28, 44]. Although a close relationship between property testing and properties of functions, and in the context of learning has been known since the beginning of the area (these topics were explored as early as in [46, 29]), *the setting presented in this paper has received little attention.* The classical framework of testing properties of functions typically assumes an oracle access to the functions and/or the inputs rather than a computational device, see [9, 28]. (For example, most function property testers assume oracle access to the values of a given function on a specific input, and typically, graph testers assume access to the edges of the input graph.) Similarly, one normally studies networks in the context of their combinatorial properties rather than by analyzing "functions" they compute. The only work that we are aware of that considers a model similar to ours is the analysis of testing mixing property of Markov chains by Batu et al. [7]. We are not aware of any property testing work for basic logical or arithmetic circuits (even a related testing satisfiability in [2] uses a different setting), though some works on testing branching programs [25, 38] or whether a function can be represented by a small/shallow circuit/branching program [22, 24, 42], are of similar flavor. Other related works include property testing of some matrix properties (e.g., [5], where testing the rank of a matrix A has implications for some properties of Ax), but these works use a different setting, rely on linear operators, and do not apply to non-linear operators like those studied in our paper.

Neural networks. The complexity of neural networks has received significant attention in the past. In particular, threshold circuits have been studied as a model of computation and for corresponding complexity class TC^0 we know $\mathsf{AC}^0 \subsetneq \mathsf{TC}^0 \subseteq \mathsf{NC}^1$ [50]. The problem of verifying whether there exist weights such that an input circuit computes an input Boolean

function (threshold circuit loading) is known to be NP-complete [33], even when the circuit is restricted to three neurons [11]. For more details on the computational complexity of networks, see, e.g., [40].

In general, feedforward networks (acyclic networks) are universal approximators when viewed as computing a continuous function, i.e., they can arbitrarily well approximate any Borel-measurable function between finite dimensional spaces provided a sufficient number of neurons (nodes) in the hidden layer exist [32]. Furthermore, width-bounded networks (but generally with a large number of layers) are universal approximators [35]. The VC-dimension of neural networks with linear threshold gates has been studied and there is an upper bound of $O(m \log m)$ where m is the number of linear threshold gates [8]. These results imply bounds on the number of samples required to train the network. However, for our setting they do not seem to be directly relevant since one cannot evaluate a training example without querying the whole network.

In a related study, [20] investigated the problem of deciding if a neural network with real inputs is monotone and [37] considered monotone neural networks.

There has been extensive research on the computational complexity of training constant depth ReLU networks. It is known that even for a single ReLU unit, the problem of finding weights minimizing the squared error of a given training set is hard to approximate [27, 36, 21] (cf. [12] for a slightly different proof/result). Further results show that even for a single neuron, the training problem parameterized by the dimension d of the training data is $W[1]$-hard and there is a $n^{\Omega(d)}$ conditional lower bound assuming the exponential time hypothesis [26]. Studies have also explored learning linear combinations of ReLUs from Gaussian distributions (e.g., an algorithm for a constant number of ReLU units has been recently developed in [14]).

2 Property testing model

We begin our study with a feedforward network with one hidden layer and ReLU activation function. The network (see Figure 1) is fully connected between the layers, and has n input nodes labeled $\{1, \ldots, n\}$, a hidden layer with m nodes labeled $\{1, \ldots, m\}$, and one output node. (We later generalize our model to multiple hidden layers and output nodes, see Section 3.4.) We associate a *binary vector* $x = (x_1, \ldots, x_n)$ with the input nodes, so that input node i has value x_i. Between the input nodes and the hidden layer there is a *complete bipartite graph with real edge weights* from $[-1, 1]$. We define a_{ij} to be the weight between input node i and node j of the hidden layer. The *value of an input node is* x_i. The *value computed at a hidden layer node* j *is* $\sum_i a_{ji} x_i$. Thus, we can write the values at the hidden layer by a matrix multiplication Ax for an $m \times n$ matrix A. To the values at the hidden layer, we apply the *ReLU activation function*, which is defined as $\mathrm{ReLU}(y) := \max\{0, y\}$. For a vector $y = (y_1, \ldots, y_m)^T$, we define $\mathrm{ReLU}(y)$ to be $\mathrm{ReLU}(y) := (\mathrm{ReLU}(y_1), \ldots, \mathrm{ReLU}(y_m))^T$. We can now write the output of the hidden layer nodes as $\mathrm{ReLU}(Ax)$. Between the hidden layer and the output node there is a complete bipartite graph with real weights from $[-1, 1]$. Let w_j denote the weight of the edge connecting hidden layer node j to the output node. If $z = \mathrm{ReLU}(Ax)$ is the vector of hidden layer outputs, then $w^T z$ is the *value of output node*, where $w = (w_1, \ldots, w_m)^T$. Thus, on input $x \in \{0, 1\}^n$ the value at the output node is $w^T \cdot \mathrm{ReLU}(Ax)$. For further intuition, we refer the reader to Figure 1.

ReLU network as a binary classifier. We interpret the network as a *binary classifier*: the network assigns each input vector to one of two classes, defined as 0 and 1. If the value at the output node is at most 0 we assign class 0, if it is larger than 0 we assign class 1. In this paper

we focus on the setting when $x \in \{0,1\}^n$ is a binary vector. Thus, we view the neural network as a function $f : \{0,1\}^n \to \{0,1\}$ that computes 1 if the output value of the network is greater than 0, and 0 if the output value is at most 0. We write[3] $f(x) = \mathrm{sgn}(\mathrm{ReLU}(w^T \cdot \mathrm{ReLU}(Ax)))$. For a given input $x \in \{0,1\}^n$, we call $f(x)$ the *output of the network*.

Our goal is to understand how the local structure of the network is related to the function f or to properties of that function. We will address this question by studying the function from a *property testing* point of view. This leads us to the following definition.

▶ **Definition 1 (ReLU network with *one hidden layer* and *single output*).** *A ReLU network with n input nodes, m hidden layer nodes, and one output node is a pair (A, w), where $A \in [-1,1]^{m \times n}$ and $w \in [-1,1]^m$. The binary function $f : \{0,1\}^n \to \{0,1\}$ computed by the ReLU network (A, w) is defined as*

$$f(x) := \mathrm{sgn}(\mathrm{ReLU}(w^T \cdot \mathrm{ReLU}(Ax))).$$

2.1 Property testing: ReLU networks with *one hidden layer/one output*

We introduce a new property testing model for computational networks on an example of neural ReLU networks. We view (see Figure 1) the network as a weighted graph $G = (V, E)$, where $V = I \cup H \cup O$ with I being the set of *input nodes*, H the set of *hidden layer nodes*, and O the set of *output nodes* (frequently $|O| = 1$). The nodes in I, H and O are labelled from 1 to $|I|, |H|$ and $|O|$, respectively. Furthermore, E contains all edges between I and H, and between H and O. The edges between I and H are also referred to as *first layer edges* and the edges from H to O as *second layer edges*. We assume that we have *query access to the edge weights*: we can specify nodes number i in I and number j from H, and query the value a_{ij} of that edge in $O(1)$ time. Similarly, we can access the entries of w by specifying the corresponding hidden layer node.

Our objective is to design (randomized) sampling algorithms that decide whether a given neural network has a certain property or is far away from the property. To achieve this goal, we must first define a distance measure of neural networks. We view the network as a weighted graph and use the (relative) edit distance. However, since we want to take into consideration that there is a different number of edges on each layer, *for two networks (A_1, w_1) and (A_2, w_2)* with n input nodes, m hidden layer nodes, and one output node, we *define their distance* as $\max\left\{\frac{\|A_1 - A_2\|_0}{mn}, \frac{\|w_1 - w_2\|_0}{m}\right\}$, where we use the standard notation $\|\cdot\|_0$ to denote the number of non-zero entries in a matrix/vector.

In this paper, we view neural networks as computing Boolean functions and study the question whether a neural network computes a function that has a certain *property*. We begin with the study of networks that compute a specific function. Since there are often many networks that compute the same function, the question of whether a given network computes a certain function may be viewed as testing whether an input has a certain specific property in standard property testing.

Given a function $f : \{0,1\}^n \to \{0,1\}$, a *property tester* is an algorithm with query access to the weights of a network (with n input nodes and m hidden layer nodes) that accepts with probability at least $\frac{2}{3}$ every network that computes f and rejects with probability at least $\frac{2}{3}$

[3] Note: the outer ReLU is *not redundant* in $\mathrm{sgn}(\mathrm{ReLU}(\cdot))$ because the argument $w^T \cdot \mathrm{ReLU}(Ax)$ can be negative, in which case $\mathrm{sgn}(\cdot)$ would return $-1 \notin \{0,1\}$; see also the discussion in footnote 1.

every network that is *far* from computing f, where we parameterize the notion of being far with parameters ε, δ. Parameter ε refers to the distance between the networks and δ refers to the distance between the computed function with respect to the uniform distribution[4].

The model. Let us recall the notion of *property of Boolean functions*, which is any family $\mathcal{P} = \bigcup_{n\in\mathbb{N}} \mathcal{P}_n$, where $\mathcal{P}_n$ is a set of Boolean functions $f : \{0,1\}^n \to \{0,1\}$.

▶ **Definition 2 (ReLU network far from a property of functions).** *Let (A, w) be a ReLU network with n input nodes and m hidden layer nodes. (A, w) is called* ***(ε, δ)-close to computing a function $f : \{0,1\}^n \to \{0,1\}$ with property $\mathcal{P} = \bigcup_{n\in\mathbb{N}} \mathcal{P}_n$****, if one can change the matrix A in at most εnm places and the weight vector w in at most εm places to obtain a ReLU network that computes a function g, such that there exists a function $f \in \mathcal{P}_n$ for which $\Pr[g(x) \neq f(x)] \leq \delta$, where x is chosen uniformly at random from $\{0,1\}^n$. If (A, w) is not (ε, δ)-close to computing f with property $\mathcal{P}$ then we say that it is* ***(ε, δ)-far from computing f with property $\mathcal{P}$****.*

If $|\mathcal{P}_n| = 1$ and $f \in \mathcal{P}_n$, then in Definition 2 we say that a *network is (ε, δ)-close/far from computing f.*

▶ Remark 3. It might seem highly undesirable that the notion of (ε, δ)-distance employed in this paper does not satisfy the triangle inequality. However, our model is a natural relaxation of the model with $\delta = 0$, which is exactly the standard model of property testing with a property $\mathcal{P}$ being the set of networks computing a given function (like the 0-function) [28]. *It is an inherent feature of property testing that the distance to a property/function often does not satisfy the triangle inequality*: Just as two graphs G and H being close to 3-colorable does not imply that G and H are close to each other, or two networks close to computing the 0-function are not necessarily close.

▶ **Definition 4.** *A property $\mathcal{P} = \bigcup_{n\in\mathbb{N}} \mathcal{P}_n$ of Boolean functions is* ***testable*** *(for ReLU networks)* ***with query complexity $q(\varepsilon, \delta, n, m)$****, if for every $n, m \in \mathbb{N}$, every $\varepsilon \in (0,1)$, and every $\delta \in (0,1)$ there exists an algorithm that receives parameters $n, m, \varepsilon, \delta$ as input and is given query access to an arbitrary ReLU network with n input nodes, m hidden layer nodes and one output node, and that*

- *queries the ReLU network in at most $q(\varepsilon, \delta, n, m)$ many places,*
- *accepts with probability at least $\frac{2}{3}$ every network that computes a function from $\mathcal{P}_n$, and*
- *rejects with probability at least $\frac{2}{3}$ every network that computes a function (ε, δ)-far from $\mathcal{P}$.*

If the function q depends only on ε and δ, then we just say that the property $\mathcal{P}$ is ***testable****.*[5]

A tester that accepts every function from $\mathcal{P}$ with probability 1 is called ***one-sided tester****.*

---

[4] While distance to a property is often parameterized using only the parameter ε, a relaxation with another parameter has been studied before. In fact, a very similar model has been used in [7] to test if a Markov chain is mixing. Similarly to our problem, the graph/Markov chain can be seen as a computational model parameterized in terms of ε and δ, where ε refers to the edit distance of the Markov chain and δ to the ℓ_1-distance between the t-step distributions. We believe that a similar relaxation is useful in our case as the relation between the network and the function is rather complex, and we have little hope to obtain results without this second parameter δ.

[5] In property testing, traditionally (cf. [9, 28]), the ultimate goal of the analysis is to *obtain testers whose query complexity is independent of the input size*, depending only on ε, δ. Testers with query complexity $o(mn)$ are of interest, but often not as the main objective. The importance of having the query complexity independent of ε and δ is especially amplified in our setting: if we query the ReLU network in $q(\varepsilon, \delta, n, m)$ many places, then normally the tester will be computing the function on the subnetwork determined by the tested edges *for all* 2^n *inputs*. However, the number of *relevant* input bits is always upper bounded by $q(\varepsilon, \delta, n, m)$ (since only input bits adjacent to the queried edges impact the function computed), and therefore if the query complexity is independent of the input size, then the number of inputs to be considered is at most $2^{q(\varepsilon,\delta)}$, which is independent on n and m. Hence, *testers with query complexity independent of n and m have their running time independent of n and m* too.

3 Technical overview

We initiate the study of property testing of computational networks on an example of neural networks. Below we presented a detailed discussion of our findings and of the technical challenges involved in our analysis. Due to space constraints, we omit the full technical details for the claims made here. All remaining proofs and a comprehensive analysis are deferred to the full version [19], except for one key result proven in Section 4.

3.1 Studying variants of the model on a simple function

We begin with the study of the constant 0-function (the same results are true for the OR function) in our basic model of feedforward *networks with ReLU activation function, one hidden layer, and single output*. We start by showing that the constant 0-function is testable in such model. This is the case even though, as we can easily show, the problem of verifying if the network computes the constant 0-function is NP-hard (note that if we had no ReLU we could simply compute $w^T A$ and check whether $w^T A$ has positive entries; so here ReLU makes the problem more difficult); see [19] for a formal proof. Our result is summarized in the following theorem (for a proof of part (1), see Section 4; for a proof of part (2) see [19]):

▶ **Theorem 5.** *Let (A, w) be a ReLU network with n input nodes, m hidden layer nodes, and a single output. Let $\delta \geq e^{-n/16}$, $\frac{1}{m} < \varepsilon < \frac{1}{2}$, and $0 < \lambda < \frac{1}{2}$.*

(1) *There is a tester that queries $O(\frac{\ln^2(1/\varepsilon\lambda)}{\varepsilon^6})$ entries from A and w, and (i) accepts with probability at least $\geq 1-\lambda$, if the ReLU network (A, w) computes the constant 0-function, and (ii) rejects with probability at least $1-\lambda$, if (A, w) is (ε, δ)-far from computing the constant 0-function.*

(2) *There is a tester that queries $O(\frac{\ln^2(1/\varepsilon\lambda)}{\varepsilon^6})$ entries from A and w, and (i) accepts with probability at least $1-\lambda$, if the ReLU network (A, w) computes the OR-function, and (ii) rejects with probability at least $1-\lambda$, if (A, w) is (ε, δ)-far from computing the OR-function.*

The query complexity of our tester does not depend on δ and we only require $\delta \geq e^{-\Omega(n)}$. *Assuming that n and m are polynomially related, one could also remove this dependence* by introducing a factor $\mathrm{polylog}(1/\delta)$ in the complexity, since for $\delta < e^{-n/16}$ we can read the whole network.

Observe that the *query complexity of our tester in Theorem 5 is independent of the network size*. Further, for constant ε and λ, the query complexity of our tester is $O(1)$.

Our property tester for the constant 0-function samples a subset of $\mathrm{poly}(1/\varepsilon)\cdot\mathrm{polylog}(1/\lambda)$ input and hidden layer nodes and evaluates whether the induced network with a *suitable bias* at the output node computes the constant 0-function. While this approach seems natural, the analysis and *its use of the bias* is non-standard.

The first step of our analysis shows that if the network is (ε, δ)-far from computing the constant 0-function then there is an input $x_0 \in \{0,1\}^n$ on which the network outputs value $\Omega(\varepsilon nm)$. Then, we show that sampling a constant number of hidden layer nodes approximates the scaled output value for this particular x_0 with sufficient accuracy, giving the scaled output value $\Omega(\varepsilon nm)$. In the next step, we show that we can sample a constant number of input nodes and approximate the value at the sampled hidden layer nodes so that the value at the output node is positive. Combining all the steps yields that if the network is (ε, δ)-far then our tester rejects with sufficient probability.

In order to show that the algorithm accepts a network computing the constant 0-function, we will argue that for all inputs $x \in \{0,1\}^n$ the sampled network computes the constant 0-function. For that, we take a different view and consider the sample from the input nodes

as being fixed. We then observe that restricting our network to the sampled input nodes can also be done by restricting the input to vectors $x \in \{0,1\}^n$ that are 0 for all input nodes that do not belong to the sample. This allow us to interpret our sampling approach as only sampling from the hidden layer. Hence, we can apply a concentration bound and take a union bound over all choices of input vectors that are zero on the nodes not in the sample. We also observe that while the analysis considers n-dimensional input vectors we know that these are 0 except for the sampled input nodes. Thus, in order to compute the output value we only need to query edges between sampled nodes.

Comparison to a vanilla testing algorithm. The above result implies that a network far from computing the 0-function (or from computing the OR-function) already exhibits this property across most of its small subnetworks. While this may not seem to be too surprising in the first place, it turns out to be a useful addition to a *vanilla testing algorithm* that *samples inputs uniformly at random* and *evaluates the network on these inputs*. This vanilla testing algorithm is arguably the most natural and routinely used approach to test the functionality of neural networks. In fact, it corresponds to the most *standard setting in property testing of functions* [28, 9] in which *the oracle allows to query the value of the network on a given input* x, rather than to query the network's weights. In our paper (see [19], we demonstrate the versatility of our property testing model and show that there are networks that *can be recognized as* (ε,δ)*-far* (for constant ε and exponentially small δ) from computing the 0-function *by our tester*, but the *vanilla tester in expectation requires an exponential in* n *number of samples* in order to find an input that does not evaluate to 0.

▶ **Theorem 6.** *Let* $n \in \mathbb{N}$, $0 < \varepsilon < \frac{1}{1000}$, $\delta < 2^{-(16\sqrt{\varepsilon}n+1)}$, *and* $\sqrt{\varepsilon}n$ *be integral.*

(1) *There is a ReLU network* (A_L, w_L) *with* n *hidden layer nodes and* n *input nodes such that with probability at most* $e^{-10\sqrt{\varepsilon}n/8}$, *a vector* x *chosen uniformly at random from* $\{0,1\}^n$ *satisfies* $w_L^T \cdot \mathrm{ReLU}(A_L x) > 0$.

(2) *The ReLU network* (A_L, w_L) *is* (ε,δ)*-far from computing the constant* 0*-function.*

Testing with one-sided error. We also consider the testability of the constant 0-function and the OR-function with *one-sided error*. We design a tester with one-sided error and query complexity of $\widetilde{O}(m/\varepsilon^2)$. The one-sided error property tester samples a subset of $O(\log m/\varepsilon^2)$ input nodes S and rejects only if there exists an input that is zero on all nodes outside of S and that evaluates to 1 or 0, respectively. If such an input is found, then it is a counter-example to computing the constant 0-function (or the OR-function, respectively) and the algorithm can reject.

Next, we give a near matching lower bound: any tester (for any constant ε) for the constant 0-function (or for the OR-function) that has one-sided error has query complexity of $\Omega(m)$. The lower bound follows from the fact that if too few weights are known, one can always "complete" the network to enforce it to compute the constant 0-function. Thus, the tester always accepts, but there are inputs that are (ε,δ)-far from computing the 0-function.

▶ **Theorem 7.**

(1) *Let* (A,w) *be a ReLU network with* n *input nodes and* m *hidden layer nodes. Let* $e^{-n/16} \leq \delta < 1$, $\frac{1}{m} < \varepsilon < \frac{1}{2}$ *and* $0 < \lambda < \frac{1}{2}$. *There is a tester that queries* $O(\frac{m \cdot \ln(m/\lambda)}{\varepsilon^2})$ *entries from* A *and* w *and always accepts, if the ReLU network* (A,w) *computes the constant* 0*-function, and rejects with probability at least* $1-\lambda$, *if the ReLU network* (A,w) *is* (ε,δ)*-far from computing the constant* 0*-function. Furthermore, the same result holds for the OR-function.*

(2) *Any tester for the constant* 0*-function (or for the OR-function) that has one-sided error has query complexity of* $\Omega(m)$.

Why testing of the 0-function? We consider the constant 0- and the OR-functions to be representative for natural, simple, yet rich, and highly non-trivial functions in our setting. Furthermore, note that for functions f and g, $f \equiv g$ iff the network $\mathrm{ReLU}(f-g)+\mathrm{ReLU}(g-f)$ computes the 0 function, and therefore testing the 0-function is closely related to the question of two networks computing the same output values for all inputs. Other related problems that might be in reach are dictatorship and juntas (which look like essentially equivalent to 0-testing, but they are not).

3.2 Lower bound in a *distribution-free model* (or on the model's choice)

It is natural to try to extend our framework to a *distribution-free model* (see, e.g., [29, Section 2], [28, Chapter 12.4]), where the notion of being (ε, δ)-far from a property defines the farness with respect to the second parameter δ in terms of *arbitrary distribution*. In the distribution-free model the distance between Boolean functions is measured with respect to an unknown distribution rather than the uniform distribution, as in our model considered above. It is also assumed that a property tester has sample access to this distribution. In our case, we will assume *query access to bits of the sample* since our goal is to be sublinear in the input size n or even to have a constant query complexity. While the study of this model is very tempting, unfortunately, we show a *lower bound that for any constant $c < 1$, rules out $O(n^c)$-time testers even for the constant* 0 *function*:

▶ **Theorem 8.** *For every constant $k > 1$, every property tester for the constant* 0*-function in the distribution-free model has a query complexity of $\Omega(n^{1-1/k})$.*

We will sketch here the main idea of the proof for $k = 2$ (where for purpose of exposition we use different constants than in the general proof, see [19]). We define two distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ on pairs of a network and an input distribution such that the networks from the first distribution are (whp.) computing the constant 0-function and such that the pairs of network and input distribution $\mathcal{D}_I$ from the second distribution are (whp.) (ε, δ)-far from computing the constant 0-function with respect to $\mathcal{D}_I$.

For $\mathcal{D}_1$ the network will (whp.) compute the constant 0-function, so the underlying distribution is not relevant. To construct the network we partition the hidden layer nodes into sets N and P such that the nodes in N are connected to the output by edges of weight -1 and the nodes in P by edges of weight 1. In the first layer, the edges to nodes from N are 1 with probability $\frac{2}{3}$ and -1, otherwise, and the edges to nodes from P are 1 with probability $\frac{1}{2}$. Ignoring the effect of the ReLU it is not too hard to see that the network will evaluate (whp.) to 0 on any fixed input and one can apply a union bound to show that this holds also for all inputs. As we show in the proof, this intuition still holds under the ReLU function.

Networks from $\mathcal{D}_2$ are similar, except that there is a random matching of input nodes and the connections to the nodes from P are sampled for one node of the matching pair and copied for the other. This implies that with probability $\frac{1}{2}$ the pair of nodes contributes 2 to the hidden layer node, while the expected contribution to N is $\frac{8}{9}$ (here the edge weights are sampled independently and the matched pair of input nodes contributes only if both first layer edges are 1, which happens with probability $\frac{4}{9}$ and the contribution is 2). Thus, for any matching pair the expected contribution to the output is linear in the size of the hidden layer. We now define the unknown input distribution to be the uniform distribution on the pairs of matched inputs. The network is far from computing the 0-function since "fixing" one of the $\Omega(n)$ matched pairs requires $\Omega(m)$ changes in the network.

Finally, we observe that distinguishing between the two distributions is expensive as the answers to the queries are identically distributed as long as no matched pair is queried. Having access to samples from the distribution does not help as it is impossible to "find"

input bits that are 1. Since the matching of input bits is random, we prove that one needs $\Omega(\sqrt{n})$ samples to find a matching pair. We formalize this intuition and parameterize it by replacing pairs of inputs by k-sets, where the first $k-1$ inputs are chosen randomly with probability $\frac{1}{2}$ to be 1 (and -1, otherwise) and the last one is 1, if there is an odd number of 1s among the first $k-1$. We then modify our construction to get a lower bound for any k. The lower bound is one of our main technical contributions, relying on a novel controlling of the expectation of ReLU of sums of binomially distributed random variables.

3.3 First steps towards a classification of testable properties?

Our next goal is to try to understand *which properties of ReLU networks are testable*. Our first result shows that some classes of functions, *symmetric Boolean functions* or *monotone Boolean functions* have trivial testers (that always accept). We prove this by showing that every network is close to computing the constant 0 function or is close to the OR-function (with ε, δ not too small).[6]

▶ **Theorem 9.** *Let (A, w) be a ReLU network with n input nodes, m hidden layer nodes, and a single output. Let $0 < \delta \leq \frac{1}{2}$ and $\max\{\frac{1}{m}, 2\sqrt{\log(2/\delta)/n}\} \leq \varepsilon$. Then (A, w) is (ε, δ)-close to computing the constant 0-function or it is (ε, δ)-close to computing the OR-function.*

The proof first argues that the value at the output (before we apply sgn and ReLU) typically deviates from its expectation by at most $O(\sqrt{nm})$. Then we show that we can modify the network in an ε-fraction of its inputs and increase (or decrease) the expected value at the output bit by more than $\frac{1}{2}\varepsilon mn$. These two facts can be shown to yield Theorem 9.

Note that the property of monotone functions includes both the 0-function *and* the OR-function. Thus an immediate consequence of Theorem 9 is that *monotonicity has a trivial tester* that always accepts if $\varepsilon \geq \max\{\frac{1}{m}, 2\sqrt{\log(2/\delta)/n}\}$, and reads the whole input if $\varepsilon < \max\{\frac{1}{m}, 2\sqrt{\log(2/\delta)/n}\}$. When n and m are polynomially related, the query complexity of the tester is $\mathrm{poly}(1/\varepsilon) \cdot \mathrm{polylog}(1/\delta)$.

Finally, we study *monotone properties* (which are different objects than monotone functions). A property $\mathcal{P}$ is *monotone*, if it is closed under flipping bits in the truth table of any function having the property $\mathcal{P}$ from 0 to 1. One can think of monotone properties as being defined by the set of minimal functions under the operation of flipping zeros to ones. We call such a set a *generator*.

We show that every monotone property is testable with query complexity logarithmically on the size of the generator (we assume that the algorithm gets the generator as input; this assumption is justified as this depends on the property only). We prove the following result.

▶ **Theorem 10.** *Let $\frac{1}{m} < \varepsilon < 1$, $2e^{-n/16} \leq \delta \leq 1$. Let $\mathcal{P} = \bigcup_n \mathcal{P}_n$ be an $f(n)$-generatible monotone property with generator $G = \bigcup_n G_n$ with $|G_n| \leq f(n)$ for all n. There is a property tester with query access to a ReLU network $\mathcal{N}$ with n input nodes and m hidden layer nodes that (i) accepts with probability at least $1-\lambda$, if $\mathcal{N}$ computes a function in $\mathcal{P}$, (ii) rejects with probability at least $1-\lambda$, if $\mathcal{N}$ is (ε, δ)-far from computing a function in $\mathcal{P}$. The tester has query complexity $O(\ln(\frac{f(n)}{\delta\varepsilon\lambda})/(\delta\varepsilon^4))$.*

[6] We would like to stress that if two networks are close to computing a given function, then *they are not necessarily* pairwise close to each other (cf. Remark 3). Similarly, notice that while the closeness result in Theorem 9 may on the first glance seem to be unnatural, similar and even stronger claims are known in various scenarios in property testing. For example, in the classical dense graph model (see [28]), for $\varepsilon > \frac{1}{n}$, every graph is *both*, ε-close to be connected and ε-close to be disconnected (every graph on n vertices can be made connected using at most n edge changes, and can be made disconnected using at most n edge changes). Therefore, the fact that in our property testing framework all properties are close to computing the constant 0-function or close to computing the OR-function, does not mean that testing of these two properties is trivial, nor it shows that the framework considered here is unnatural.

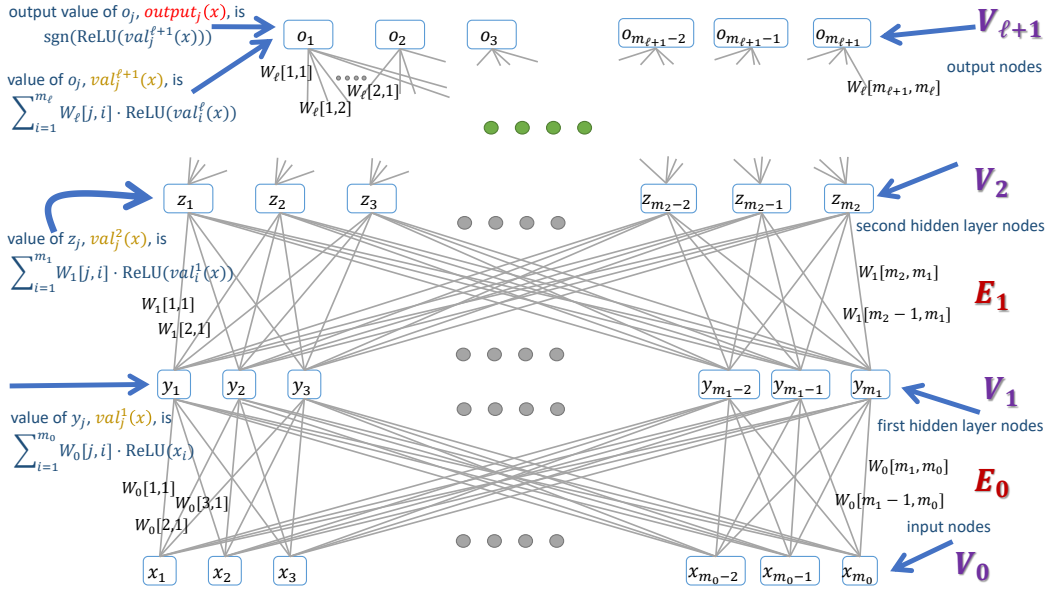


Figure 2 A general ReLU network with multiple inputs, multiple outputs, and multiple layers, extending the network depicted on Figure 1. ReLU network $(W_0, \ldots, W_\ell)$ has $\ell + 2$ layers:

- layer zero V_0 containing m_0 *input nodes*,
- ℓ *hidden layers* $1, \ldots, \ell$, with layer $1 \le k \le \ell$ containing m_k nodes V_k, and
- layer $(\ell + 1)$ containing $m_{\ell+1}$ *output nodes* $V_{\ell+1}$.

Weights: The network has *weighted edges* connecting pairs of consecutive layers. For every $0 \le k \le \ell$ and for every $1 \le i \le m_k$ and $1 \le j \le m_{k+1}$, there is an edge with weight $W_k[j,i] \in [-1,1]$ connecting the i-th node at layer k with the j-th node at layer $k+1$.
Values: For a given input $x = (x_1, \ldots, x_{m_0})^T \in \{0,1\}^{m_0}$, for any node j at layer 0, the *value* of that node $\mathrm{val}_j^0(x) = x_j$ (which is also the j-th row of $f_0(x)$). For any node j at layer $1 \le k \le \ell + 1$, the *value* of that node is $\mathrm{val}_j^k(x) = \sum_{i=1}^{m_{k-1}} W_{k-1}[j,i] \cdot \mathrm{ReLU}(\mathrm{val}_i^{k-1}(x))$; notice that this is the j-th row of $f_k(x)$.
Output: For any node j at layer $\ell + 1$ (i.e., for the j-th output node), the binary *output* computed by that node equals $\mathrm{output}_j(x) = \mathrm{sgn}(\mathrm{ReLU}(\mathrm{val}_j^{\ell+1}(x)))$; notice that this is the j-th row of $f(x)$.

3.4 Extension to ReLU networks with *multiple hidden layers/outputs*

For simplicity of presentation, we focused here mainly on the model for ReLU networks only with one output and one hidden layer. Our study, however, can be naturally extended to ReLU networks with multiple outputs and multiple hidden layers (see also Figure 2).

▶ **Definition 11 (ReLU networks with multiple hidden layers and multiple outputs).** *A ReLU network with n input nodes, $\ell \ge 1$ hidden layers with $m_1, \ldots, m_\ell$ hidden layer nodes each, and r output nodes is a tuple $(W_0, W_1, \ldots, W_\ell)$, where $W_i \in [-1,1]^{m_{i+1} \times m_i}$ for every $0 \le i \le \ell$, with $m_0 = n$ and $m_{\ell+1} = r$. The function $f : \{0,1\}^n \to \{0,1\}^r$ computed by a ReLU network $(W_0, \ldots, W_\ell)$ is defined recursively as $f(x) := \mathrm{sgn}(\mathrm{ReLU}(f_{\ell+1}(x)))$,*

$$f_i(x) := \begin{cases} W_{i-1} \cdot \mathrm{ReLU}(f_{i-1}(x)) & \text{if } 1 \le i \le \ell + 1, \\ x & \text{if } i = 0. \end{cases}$$

Observe that unfolding the recurrence in Definition 11 gives a non-recursive version of the value $f(x)$ computed by the network (in a form resembling Definition 1) is

$$f(x) := \mathrm{sgn}(\mathrm{ReLU}(W_\ell \cdot \ \ldots \ \cdot \mathrm{ReLU}(W_2 \cdot \mathrm{ReLU}(W_1 \cdot \mathrm{ReLU}(W_0 \cdot \mathrm{ReLU}(x)))) \ldots)).$$

One can also easily extend Definitions 2 and 4 to ReLU networks with multiple hidden layers and multiple outputs (for more details, see the full version of the paper [19]).

We can extend our framework to more complex neural networks and demonstrate that our methodology can be applied to a natural generalization of our setting allowing *multiple outputs* and *multiple hidden layers*. We show the applicability of our techniques by extending the approach for testing the constant 0- and the OR-function to testing near-constant functions in ReLU networks with multiple hidden layers and outputs: testing for $f(x) = \mathrm{sgn}(\mathrm{ReLU}(W_\ell \cdot \ \ldots \ \cdot \mathrm{ReLU}(W_0 \cdot \mathrm{ReLU}(x)) \ldots))$, where $W_i \in [-1,1]^{m_{i+1} \times m_i}$ and $x \in \{0,1\}^{m_0}$.

We begin with the analysis of ReLU networks with multiple outputs and *one hidden layer*. Before we proceed, let us introduce the notion of *near-constant functions* with multiple outputs[7], which are constant functions on every input x except $x = \mathbf{0}$. We first present a general reduction to single layer networks. The following theorem, central for our analysis, demonstrates that if a ReLU network with r outputs is (ε, δ)-far from computing near-constant function $\mathbf{b}$, then for at least an $\frac{\varepsilon}{2}$-fraction of the output bits i, the ReLU network restricted to the output node i is (ε', δ')-far from computing an appropriate near-constant function, where $\delta' \sim \delta/r$ and $\varepsilon' = \varepsilon^2/1025$.

▶ **Theorem 12.** *Let $\mathcal{N}$ be a ReLU network (A, W) with n input nodes, m hidden layer nodes, and r output nodes. Let $e^{-n/16} \leq \delta < 1$ and $16 \cdot \sqrt{\frac{\ln(2m)}{m}} \leq \varepsilon < \frac{1}{2}$. Let $\mathbf{b} = (\mathfrak{b}_1, \ldots, \mathfrak{b}_r) \in \{0,1\}^r$. If $\mathcal{N}$ is (ε, δ)-far from computing a near-constant function $\mathbf{b}$ then there are more than $\frac{1}{2}\varepsilon r$ output nodes such that for any such output node i, the ReLU network $\mathcal{N}$ restricted to the output node i is $(\frac{\varepsilon^2}{1025}, \frac{\delta - e^{-n/16}}{r})$-far from computing near-constant function $\mathfrak{b}_i$.*

One can then combine Theorems 5 and 12 to obtain the following theorem for testing near-constant functions in ReLU networks with *multiple outputs and one hidden layer*.

▶ **Theorem 13.** *Let $\mathcal{N}$ be a ReLU network (A, W) with n input nodes, m hidden layer nodes, and r output nodes. Let $\mathbf{b} \in \{0,1\}^r$, $(r+1)e^{-n/16} \leq \delta < 1$, and $c \cdot \sqrt{\ln(2m)/m} < \varepsilon < \frac{1}{2}$ for a sufficiently large constant c. There is a property tester that queries $O(\frac{\ln^2(1/\varepsilon\lambda)}{\varepsilon^{13}})$ entries from A and W and*

- *accepts with probability at least $\frac{2}{3}$, if $\mathcal{N}$ computes near-constant function $\mathbf{b}$, and*
- *rejects with probability at least $\frac{2}{3}$, if $\mathcal{N}$ is (ε, δ)-far from computing near-constant function $\mathbf{b}$.*

Observe that the *query complexity of our tester in Theorem 13 is independent of the network size*. Further, for constant ε and λ, the query complexity is $O(1)$, and similarly as in Theorem 5, one could remove the dependency for δ by introducing a small factor term in the query complexity.

Our analysis so far has studied the model of ReLU networks with a single hidden layer. We can extend our analysis of this setting to allow *multiple hidden layers*. A simplified claim is as follows.

[7] We need this notion to exclude trivial cases; notice that since for any ReLU network $f(\mathbf{0}) = \mathbf{0}$ (and so there is no ReLU network with $\forall_x f(x) = 1$), we consider only *near-constant functions*.

▶ **Theorem 14.** *Let* $0 < \lambda < \frac{1}{\ell+1}$, $\frac{1}{n} < \varepsilon < \frac{1}{2}$, *and* $\delta \geq e^{-n/16} + e^{-(\varepsilon/2)^{\ell} n/(342(\ell+1)^2)}$. *Let* $\mathcal{N}$ *be a ReLU network* $(W_0, \ldots, W_\ell)$ *with* n *input nodes,* $\ell \geq 1$ *hidden layers with* n *hidden layer nodes each, and a single output node. Further, suppose that* $n \geq 171 \cdot (\ell+1)^2 \cdot (2/\varepsilon)^{2\ell} \cdot (\ln(2/\delta) + \ell \ln n)$. *Assuming that* ℓ *is constant, there is a tester that queries* $\Theta(\varepsilon^{-8\ell} \cdot \ln^4(1/\lambda\varepsilon))$ *entries from* $W_0, \ldots, W_\ell$, *and*

- **(i)** *rejects with probability at least* $1 - \lambda$, *if* $\mathcal{N}$ *is* (ε, δ)*-far from computing the constant* 0*-function,*
- **(ii)** *accepts with probability at least* $1 - \lambda$, *if* $\mathcal{N}$ *computes the constant* 0*-function.*

The approach used in the proof of Theorem 14 follows the method presented in the analysis of a single hidden layer, but the need of dealing with multiple layers makes the analysis significantly more complex. The approach relies on two steps of the analysis, first proving part (i) about ReLU networks that are (ε, δ)-far from computing the constant 0-function and then part (ii) about ReLU networks that compute the constant 0-function.

The analysis of part (i) is in four steps. We first show that the use of randomly sampled nodes in each layer of the network results in a modified ReLU network that, after scaling, for any fixed input returns the value close to the value returned by the original network. Next, we extend this claim to show that there is always a modified ReLU network that, after scaling, for all but a small fraction of the inputs, returns the value close to the value returned by the original network. We then study ReLU networks $(W_0, \ldots, W_\ell)$ that are (ε, δ)-far from computing the constant 0-function. In that case, we first show that there is always an input $x_0 \in \{0,1\}^{m_0}$ on which the (ε, δ)-far network returns a value which is (sufficiently) large. Then, using the result that the sampled network approximates the output of the original network, we will argue that if $(W_0, \ldots, W_\ell)$ is (ε, δ)-far from computing the constant 0-function then randomly sampled network on input x_0 returns a (sufficiently) large.

The study of part (ii) relies on a similar approach as that in part (i). We use the fact that the sampled network computes a function which on a fixed input is (after scaling) close to the original function on that input, and hence that value cannot be too large. An important difference with part (i) is that one must prove the claim to *hold for all inputs*, making the arguments more complex (the straightforward arguments would require the size of the sample to depend on m_0).

A consequence of Theorem 14 is that it gives a tester with query complexity $1/\mathrm{poly}(\varepsilon)$ for any constant number of layers. Furthermore, as in our study for one hidden layer, essentially identical analysis as for the constant 0-function can be applied to test the OR-function.

Finally, we can extend the analysis above (Theorems 13 and 14) to networks with *multiple hidden layers* and *multiple outputs*. As in Theorem 13 (for ReLU networks with a single hidden layer and multiple outputs) the proof of the following theorem relies on a reduction of the problem to testing a near-constant function to testing the 0- and the OR-function.

▶ **Theorem 15** (Testing a near-constant function)**.** *Let* $\mathcal{N}$ *be a ReLU network* $(W_0, \ldots, W_\ell)$ *with* m_0 *input nodes,* $\ell \geq 1$ *hidden layers with* $m_1, \ldots, m_\ell$ *hidden layer nodes each, and* $m_{\ell+1} \geq 2$ *output nodes. Let* $0 < \lambda < \frac{1}{\ell+1}$, $2\sqrt[\ell]{\frac{17(\ell+1)}{\max_{0 \leq k \leq \ell}\{m_k\}}} < \varepsilon < \frac{1}{2}$, *and* $\delta \geq 2m_{\ell+1}(\xi + e^{-m_0/16})$. *Let* $\mathbf{b} \in \{0,1\}^{m_{\ell+1}}$ *and* $c = c(\ell)$ *be a sufficiently large positive constant. Suppose that for an arbitrary parameter* $0 < \xi < \frac{1}{\ell+1}$ *it holds that* $m_k \geq c \cdot (2/\varepsilon)^{2\ell^2} \cdot \ln(\frac{1}{\xi} \prod_{i=1}^{\ell} m_i)$ *for every* $0 \leq k \leq \ell$. *If* ℓ *is constant then there is a tester that queries* $\Theta(\varepsilon^{-(8\ell^2+1)} \cdot \ln^4(1/\lambda\varepsilon))$ *entries from* $W_0, \ldots, W_\ell$, *and*

- *accepts with probability at least* $1 - \lambda$, *if* $\mathcal{N}$ *computes near-constant function* $\mathbf{b}$, *and*
- *rejects with probability at least* $1 - \lambda$, *if* $\mathcal{N}$ *is* (ε, δ)*-far from computing near-constant function* $\mathbf{b}$.

As with our earlier results in Section 3.4, the *query complexity of our tester in Theorem 15 is independent of the network size*, and it is $O(1)$ for constant ε, λ, ℓ, and $m_{\ell+1}$.

4 Testing constant 0-function in ReLU networks with one hidden layer

To give readers a flavor of our analysis, we present here our analysis of ReLU networks with one hidden layer and a single output – Theorem 5 (1), deferring the study of other claims and of more complex neural networks with multiple outputs and multiple hidden layers to the full version of our paper [19]. We consider the simplest function, the **constant 0-function** $f_n^{(0)} : \{0,1\}^n \to \{0,1\}$ defined as $f_n^{(0)}(x) = 0$ for all $x \in \{0,1\}^n$, and study it from a property testing point of view. We define $P_n = \{f_n^{(0)}\}$ and the property $\Pi^{(0)} = \bigcup_{n\in\mathbb{N}} P_n$. Our objective is to design a property tester for $\Pi^{(0)}$.

While one can show (see [19]) that it is hard to decide if a given ReLU network computes the constant 0-function or not, in this section we prove that a *relaxed* version of the problem, namely, that of *(property) testing* whether a given ReLU network computes the constant 0-function, has significantly lower complexity. Below we prove the first part of Theorem 5 and show that property $\Pi^{(0)}$ is testable for ReLU networks with one hidden layer.

Testing 0-function. Our property testing algorithm AllZeroTester$(\varepsilon, \lambda, n, m)$ below works by focusing its attention on a small "subnetwork" induced by randomly sampled nodes. The algorithm samples a subset of $\text{poly}(1/\varepsilon) \cdot \ln(1/\lambda)$ nodes from the input layer and from the hidden layer. Then, we scale the weights of all edges between the sampled input and the sampled hidden layer nodes by a factor of $\frac{n}{s}$ and all edges from sampled hidden layer nodes to the output node by $\frac{m}{t}$. We then introduce a *bias* $b = -\frac{1}{16}\varepsilon nm$ to the output node, i.e., we subtract $\frac{1}{16}\varepsilon nm$ before pplying the ReLU function. For the resulting network we check if it computes the constant 0-function. If this is the case, we accept; otherwise, we reject.

Algorithm 1 AllZeroTester$(\varepsilon, \lambda, n, m)$.

1 Sample $s = \lceil \frac{2^{20}}{\varepsilon^2} \cdot \ln(1/\varepsilon\lambda) \rceil$ nodes from the input layer and $t = \lceil \frac{2^{30}}{\varepsilon^4} \cdot \ln(1/\varepsilon\lambda) \rceil$ nodes from the hidden layer uniformly at random without replacement.
2 Let $S \in \mathbb{N}_0^{n\times n}$ and $T \in \mathbb{N}_0^{m\times m}$ be the corresponding sampling matrices.
3 **if** there is $x \in \{0,1\}^n$ with $\frac{nm}{st} \cdot w^T \cdot \text{ReLU}(TASx) - \frac{1}{16}\varepsilon nm > 0$ **then reject**;
4 **else accept**.

In the following, we use matrix notation and write S and T for sampling matrices that select s and t input/hidden layer nodes at random. That is, S is an $n \times n$ diagonal matrix whose diagonal entries (i,i) are equal to 1 iff x_i is in the sample set (and is 0 otherwise), and T is an $m \times m$ diagonal matrix whose diagonal entries (j,j) are equal to 1 iff hidden layer node j is in the second sample set (and is 0 otherwise). With this notation we can describe our algorithm as follows (we did not optimize constants and assume $\varepsilon \in (\frac{1}{m}, \frac{1}{2}), \lambda \in (0, \frac{1}{2})$).

Let us first observe that the query complexity of AllZeroTester is $(s+1)\cdot t = \Theta(st)$, and hence the query complexity is as stated. Indeed, after sampling s input nodes and t nodes from the hidden layer, the diagonal matrices S and T have s and t non-zero entries, respectively. Therefore TAS can be computed by querying only $s \cdot t$ entries from matrix A and it will have at most t non-zero rows and at most s non-zero columns. Hence, in order to compute $TASx$ one has to consider only s bits from $\{0,1\}^n$, and to compute $\frac{nm}{st} \cdot w^T \cdot \text{ReLU}(TASx)$ one has to consider only t entries from w^T.

Properties of the output. Next, we will study the properties of the output. First, notice that our sampling network without a bias would compute

$$\text{sgn}\left(\text{ReLU}\left(\frac{m}{t} \cdot w^T \cdot T \cdot \text{ReLU}\left(\frac{n}{s} \cdot ASx\right)\right)\right).$$

Since the ReLU function is positive homogeneous, T is a fixed matrix with non-negative entries after the sampling, $\frac{n}{s} > 0$, and $\frac{m}{t} > 0$, we have that

$$\operatorname{sgn}\left(\operatorname{ReLU}\left(\frac{m}{t}\cdot w^T\cdot T\cdot \operatorname{ReLU}\left(\frac{n}{s}\cdot ASx\right)\right)\right) = \operatorname{sgn}\left(\operatorname{ReLU}\left(\frac{nm}{st}\cdot w^T\cdot \operatorname{ReLU}(TASx)\right)\right).$$

Introducing in AllZeroTester the bias to the output node leads to a network computing

$$\operatorname{sgn}\left(\operatorname{ReLU}\left(\frac{nm}{st}\cdot w^T\cdot \operatorname{ReLU}(TASx) - \frac{1}{16}\varepsilon nm\right)\right)$$

and we notice that one can drop sgn and the outer ReLU, if we check whether the computed value is greater than 0.

4.1 One hidden layer: If (ε, δ)-far then there is a bad input

To analyze the correctness of AllZeroTester, we first present our structural lemma.

▶ **Lemma 16.** *Let $\delta \geq e^{-n/16}$ and $\frac{1}{m} < \varepsilon < \frac{1}{2}$. Let (A, w) be a ReLU network with n input nodes and m hidden layer nodes. If (A, w) is (ε, δ)-far from computing the constant 0-function then there exists an input $x \in \{0,1\}^n$ such that $w^T \cdot \operatorname{ReLU}(Ax) > \frac{1}{8}\varepsilon mn$.*

Proof. The proof of Lemma 16 is by contradiction: we will assume that for every input x the output value is at most $\frac{1}{8}\varepsilon nm$ and then we will show that the ReLU network is (ε, δ)-close to computing the constant 0-function; this would imply contradiction.

First, suppose that there are less than $\lfloor \varepsilon m \rfloor$ positively weighted edges connecting the hidden layer to the output node. Then we set all of them to 0 and the resulting network computes the constant 0-function. Thus the network is (ε, δ)-close, which is a contradiction.

Otherwise, suppose that there are at least $\lfloor \varepsilon m \rfloor$ positively weighted edges connecting the hidden layer to the output node. Select an arbitrary subset E of $\lfloor \varepsilon m \rfloor$ positively weighted edges and let U denote the set of hidden layer nodes incident to the edges from E. Let E' be the subset of first layer edges connecting input nodes to the nodes from U. Notice that $|E| = |U| = \lfloor \varepsilon m \rfloor$ and $|E'| = n \cdot |U| = \lfloor \varepsilon m \rfloor n$. Now we modify the weights of $|E| = \lfloor \varepsilon m \rfloor$ edges between the hidden layer nodes and the output, and $|E'| = \lfloor \varepsilon m \rfloor n$ edges between input nodes and hidden layer nodes: we assign weight -1 to every edge from E and weight 1 to every edge from E'.

Observe that originally, before the modifications, every hidden layer node in U had a non-negative contribution to the output. After the modifications, every hidden layer node from U contributes $-k$ to the output node, where k is the number of 1s in the input vector x, i.e., $k = \|x\|_1$. Thus all nodes from U contribute $-|U| \cdot \|x\|_1 = -\lfloor \varepsilon m \rfloor \cdot \|x\|_1$, and the changes of the weights of the edges incident to U *reduce the output value* by at least $\lfloor \varepsilon m \rfloor \cdot \|x\|_1$.

Next, let us consider a random input vector $x \in \{0,1\}^n$. Using Chernoff bounds, with probability at least $1 - e^{-n/16} \geq 1 - \delta$, we have $\|x\|_1 > \frac{1}{4}n$. Conditioned on this event, after modifying the weights in the network, the sum of inputs to the output node is *reduced* by at least $\frac{1}{4}n \cdot \lfloor \varepsilon m \rfloor \geq \frac{1}{8}\varepsilon nm$ for our range of parameters. However, we have assumed that for every input x the output value is at most $\frac{1}{8}\varepsilon nm$, and thus the network computes the constant 0-function. Hence, we modified an ε-fraction of the network's edge weights and obtained a function that disagrees with the constant 0-function only on inputs with $\|x\|_0 \leq n/4$, which is at most a δ-fraction of all inputs. This is a contradiction to the assumption that the network is (ε, δ)-far from that function. ◀

4.2 One hidden layer: Random sampling concentration

Next, we consider the impact of the random sampling matrices S and T on the output value of a ReLU network for any fixed x (and hence, for x whose existence is argued in Lemma 16).

▶ **Lemma 17.** *Let $w \in [-1,1]^m$, $A \in [-1,1]^{m\times n}$. Let S, T be sampling matrices corresponding to samples of size $t \geq \frac{2048\ln(4/\lambda)}{\varepsilon^2}$ and $s \geq \frac{2048\ln(4t/\lambda)}{\varepsilon^2}$. Then for any fixed $x \in \{0,1\}^n$*

$$\Pr\left[\left|\frac{mn}{st}\cdot w^T\cdot \mathrm{ReLU}(TASx) - w^T\cdot\mathrm{ReLU}(Ax)\right| > \tfrac{1}{16}\varepsilon nm\right] \leq \lambda\ .$$

Proof. Fix an arbitrary $x \in \{0,1\}^n$. We first show that

$$\Pr\left[\left|\frac{m}{t}\cdot w^T\cdot T\cdot \mathrm{ReLU}(Ax) - w^T\cdot\mathrm{ReLU}(Ax)\right| > \tfrac{1}{32}\varepsilon nm\right] \leq \tfrac{1}{2}\lambda\ . \tag{1}$$

Observe that $w^T\cdot T\cdot\mathrm{ReLU}(Ax)$ is the sum of t entries in $w\odot\mathrm{ReLU}(Ax)$ (here $\odot$ denotes the Hadamard product) sampled uniformly without replacement. Next, we notice that $\mathbf{E}[w^T\cdot T\cdot\mathrm{ReLU}(Ax)] = \frac{t}{m}\cdot w^T\cdot\mathrm{ReLU}(Ax)$. Rescaling by $\frac{1}{n}$ ensures that the entries in $w\odot\mathrm{ReLU}(Ax)$ are in $[-1,1]$ and so we can apply Hoeffding bounds [31] (note that the Hoeffding bound also holds for sampling without replacement [6, Proposition 1.2]) to show

$$\Pr\left[\left|\frac{1}{n}w^T T\,\mathrm{ReLU}(Ax) - \frac{t}{mn}w^T\,\mathrm{ReLU}(Ax)\right| > \tfrac{1}{32}\varepsilon t\right] =$$
$$\Pr\left[\left|\frac{1}{n}w^T T\,\mathrm{ReLU}(Ax) - \mathbf{E}\left[\frac{1}{n}w^T\cdot T\cdot\mathrm{ReLU}(Ax)\right]\right| > \tfrac{1}{32}\varepsilon t\right] \leq 2e^{-2\varepsilon^2(t/32)^2/(4t)} \leq \tfrac{1}{2}\lambda$$

for our choice of $t \geq \frac{2048\ln(4/\lambda)}{\varepsilon^2}$. Multiplying both sides with $\frac{mn}{t}$ proves inequality (1).

We can view the sampling process as first sampling t nodes from the hidden layer and then s nodes from the input layer. Thus, we can condition on T being fixed and assume that the s nodes from the input layer are sampled uniformly at random. We will now prove:

$$\Pr\left[\left|\frac{mn}{ts}\cdot w^T\cdot\mathrm{ReLU}(TASx) - \frac{m}{t}w^T\cdot T\cdot\mathrm{ReLU}(Ax)\right| > \tfrac{1}{32}\varepsilon nm\ \middle|\ T\right] \leq \tfrac{1}{2}\lambda\ . \tag{2}$$

Let a_i be the i-th row of matrix A. The value $a_i\cdot x$ is the value at the i-th hidden layer node before applying ReLU. We may view a_iSx as the sum of s random entries from $a_i\odot x$. Since these entries are from $[-1,1]$ and $\mathbf{E}[a_iSx] = \frac{s}{n}\cdot a_i\cdot x$, Hoeffding's bound implies that

$$\Pr\left[\left|a_iSx - \frac{s}{n}\cdot a_ix\right| > \varepsilon s/32\right] \leq 2e^{-2\varepsilon^2(s/32)^2/(4s)} \leq \frac{\lambda}{2t},$$

where the last inequality follows from our assumption that $s \geq \frac{2024\ln(4t/\lambda)}{\varepsilon^2}$.

Now we observe that for $w_i \in [-1,1]$ we have

$$\left|w_i\cdot\left(\mathrm{ReLU}(a_iSx) - \mathrm{ReLU}\left(\frac{s}{n}a_ix\right)\right)\right| \leq \left|w_i\cdot\left(a_iSx - \frac{s}{n}a_ix\right)\right| \leq \left|a_iSx - \frac{s}{n}a_ix\right|.$$

Following our arguments above, for each of the rows we can bound the contribution of the edge with weight w_i that connects the i-th hidden layer node to the output node to get

$$\Pr\left[\left|w_i\cdot\left(\mathrm{ReLU}(a_iSx) - \mathrm{ReLU}\left(\frac{s}{n}\cdot a_ix\right)\right)\right| > \frac{1}{32}\varepsilon s\right] \leq 2e^{-2\varepsilon^2/(s/32)^2/(4s)} \leq \frac{\lambda}{2t}.$$

Next, we observe that TA equals A in exactly t rows (the sampled rows) and is 0 otherwise. Taking a union bound over the contribution of these t rows and observing that for the remaining rows the product with Sx is 0 for every S, we obtain

$$\Pr\left[\left|w^T\cdot\mathrm{ReLU}(TASx) - \frac{s}{n}\cdot w^T\cdot\mathrm{ReLU}(TAx)\right| > \frac{1}{32}\varepsilon st\ \middle|\ T\right] \leq \frac{\lambda}{2}.$$

Multiplying with $\frac{mn}{st}$ and observing that $\mathrm{ReLU}(TAx) = T \cdot \mathrm{ReLU}(Ax)$ (since ReLU is positively homogeneous), we obtain (2):

$$\Pr\left[\left|\frac{mn}{st} \cdot w^T \cdot \mathrm{ReLU}(TASx) - \frac{m}{t} \cdot w^T \cdot T \cdot \mathrm{ReLU}(Ax)\right| > \varepsilon mn/32 \;\middle|\; T\right] \leq \frac{\lambda}{2}.$$

Next observe that inequalities (1) and (2) are simultaneously satisfied with probability at least $1 - \lambda$. Thus, the lemma follows from

$$\begin{aligned}
&\left|\frac{mn}{st} \cdot w^T \cdot \mathrm{ReLU}(TASx) - w^T \cdot \mathrm{ReLU}(Ax)\right| \leq \\
&\leq \left|\frac{mn}{st} w^T \,\mathrm{ReLU}(TASx) - \frac{m}{t} w^T T \,\mathrm{ReLU}(Ax)\right| + \left|\frac{m}{t} w^T T \,\mathrm{ReLU}(Ax) - w^T \,\mathrm{ReLU}(Ax)\right| \\
&\leq \frac{1}{16}\varepsilon mn
\end{aligned}$$

with probability at least $1 - \lambda$. ◀

4.3 One hidden layer: If (ε, δ)-far then we reject

Our next lemma incorporates Lemmas 16 and 17 to summarize the analysis of our testers for ReLU networks that are (ε, δ)-far from computing the constant 0-function.

▶ **Lemma 18.** *Let $\delta \geq e^{-n/16}$, $\frac{1}{m} < \varepsilon < \frac{1}{2}$, and $0 < \lambda < \frac{1}{2}$. Let (A, w) be a ReLU network with n input nodes and m hidden layer nodes. If (A, w) is (ε, δ)-far from computing the constant 0-function then algorithm* AllZeroTester *rejects with probability at least $1 - \lambda$.*

Proof. By Lemma 16, since (A, w) is (ε, δ)-far from computing the constant 0-function, there exists an $x_0 \in \{0,1\}^n$ such that $w^T \cdot \mathrm{ReLU}(Ax_0) > \frac{1}{8}\varepsilon mn$. Next, by Lemma 17, with probability at least $1 - \lambda$ (over the random choice of S and T in the algorithm) we have

$$\left|\frac{mn}{st} \cdot w^T \cdot \mathrm{ReLU}(TASx_0) - w^T \cdot \mathrm{ReLU}(Ax_0)\right| \leq \frac{1}{16}\varepsilon nm \ .$$

This implies that with probability $1 - \lambda$ we obtain

$$\frac{mn}{st} \cdot w^T \cdot \mathrm{ReLU}(TASx_0) - \frac{1}{16}\varepsilon mn > 0,$$

and so AllZeroTester rejects with probability at least $1 - \lambda$. This concludes the proof. ◀

4.4 One hidden layer: If network computes 0 then we accept

Our final lemma of this section summarizes the second part of the analysis of AllZeroTester for ReLU networks and deals with the networks that correctly compute the 0-function.

▶ **Lemma 19.** *Let (A, w) be a ReLU network with n input nodes and m hidden layer nodes. Let S, T be sampling matrices corresponding to samples of size $t \geq \frac{512 \ln(2^{s+1}/\lambda)}{\varepsilon^2}$ and $s \geq 1$. If (A, w) computes the constant 0-function then with probability at least $1 - \lambda$ (over the random choice of S and T as defined by the algorithm), for all $x \in \{0,1\}^n$*

$$\frac{nm}{st} \cdot w^T \cdot \mathrm{ReLU}(TASx) - \frac{1}{16}\varepsilon nm < 0,$$

and so algorithm AllZeroTester *accepts.*

Proof. Let (A, w) be a ReLU network that computes the constant 0-function. Let I_S be the set of indices of the s nodes sampled by our algorithm from the input layer and let $X_S = \{x = (x_1, \ldots, x_n)^T \in \{0,1\}^n : x_i = 0, \forall i \notin S\}$ be the set of different inputs to the sampled network. In what follows, we will condition on the choice of S being fixed. Then

$$\frac{nm}{st} \cdot w^T \cdot \mathrm{ReLU}(TASx) - \frac{1}{16}\varepsilon nm < 0$$

holds for all $x \in \{0,1\}^n$ if and only if this inequality holds for all $x \in X_S$. Therefore, in what follows we will show that the output value of a fixed $x \in X_S$ will be approximated with high probability and then we will apply a union bound over all $x \in X_S$ to conclude the analysis.

Let x be an arbitrary fixed vector from X_S. Observe that $\frac{1}{s} \cdot w^T \cdot \mathrm{ReLU}(ASx)$ is an m-dimensional vector with entries from $[-1, 1]$. We know that $w^T \mathrm{ReLU}(ASx) \leq 0$ since the network computes the constant 0-function. Since ReLU is positively homogeneous we can move T out of the ReLU function. The fact that T is a sampling matrix and that $w^T \mathrm{ReLU}(ASx) \leq 0$ yields

$$\mathbf{E}\left[\frac{1}{s} \cdot w^T \cdot T \cdot \mathrm{ReLU}(ASx) \;\middle|\; S\right] \leq 0. \tag{3}$$

Then, applying Hoeffding bound gives us

$$\Pr\left[\left|\frac{1}{s} \cdot w^T \cdot T \cdot \mathrm{ReLU}(ASx) - \mathbf{E}\left[\frac{1}{s} \cdot w^T \cdot T \cdot \mathrm{ReLU}(ASx)\right]\right| \geq \frac{1}{16}\varepsilon t \;\middle|\; S\right] \leq 2e^{\frac{-2(\varepsilon t/16)^2}{4t}},$$

which is upper bounded by $\frac{\lambda}{2^s}$ for our choice of t, since $t \geq \frac{512 \ln(2^{s+1}/\lambda)}{\varepsilon^2}$. By the union bound over the $|X_S| = 2^s$ vectors in X_S, this implies the following inequality for all $x \in \{0,1\}^n$:

$$\Pr\left[\left|\frac{mn}{st} \cdot w^T \cdot T \cdot \mathrm{ReLU}(ASx) - \mathbf{E}\left[\frac{nm}{st} \cdot w^T \cdot T \cdot \mathrm{ReLU}(ASx)\right]\right| \geq \frac{1}{16}\varepsilon nm\right] \leq \lambda.$$

Lemma 19 follows from inequality (3) and the fact that in the algorithm we are subtracting $\frac{1}{16}\varepsilon nm$ from the result by taking a union bound over the $|X_S| = 2^s$ vectors in X_S. ◀

4.5 One hidden layer: Completing the proof of Theorem 5 (1)

Now Theorem 5 (1) follows immediately from Lemmas 18 and 19 and from our analysis of the query complexity after the description of AllZeroTester on page 16. ◀

5 Conclusions

In this paper, we introduce a property testing model to study computational networks used as computational devices and illustrate our setting on a case study of simple ReLU neural networks. We believe that the property testing model developed in this paper establishes a novel and useful theoretical framework for the study of testing algorithms and offers a new and interesting viewpoint with a potential of advancing our understanding of the structure of neural networks (and, in general, of computational networks). While, due to space constraints, we have omitted full technical details behind the claims made in Section 3, we refer the reader for the proofs and a comprehensive analysis to the full version of this paper [19].

In the context of neural networks, given our poor understanding of the computational processes in modern deep networks (leading to serious questions on the trust and transparency of the underlying learning algorithms), we hope that the lens of property testing as developed in the framework considered in this paper will contribute to a better understanding of the relation of the network structure and the computed function or properties of that function.

Indeed, since the analysis of property testing algorithms relies on establishing close links between the object's local structure and the tested property, we believe the study of simple yet fundamental models of ReLU networks may provide a new insight into such networks.

Although our paper makes the first step in the analysis of ReLU networks in the context of property testing, there is still a large number of open questions (from the property testing point of view) regarding such networks, including, among others, the following:

- What is the query complexity of testing dictatorship and juntas (see [10] for a survey on testing juntas)?
- Classify the constant time testable properties with one- and two-sided error.
- Study tolerant testing [41] in our model.
- Extend the results to inputs over the reals.

These are just a small number of immediate questions one can ask about our model. We believe that there is a wide variety of interesting questions and are confident to see further research in this direction.

References

1 Noga Alon, Eldar Fischer, Ilan Newman, and Asaf Shapira. A combinatorial characterization of the testable graph properties: It's all about regularity. *SIAM Journal on Computing*, 39(1):143–167, 2009. doi:10.1137/060667177.

2 Noga Alon and Asaf Shapira. Testing satisfiability. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 645–654, 2002. URL: https://dl.acm.org/doi/10.5555/545381.545467.

3 Noga Alon and Asaf Shapira. A characterization of the (natural) graph properties testable with one-sided error. *SIAM Journal on Computing*, 37(6):1703–1727, 2008. doi:10.1137/06064888X.

4 Sunil Arya and David M. Mount. A fast and simple algorithm for computing approximate Euclidean minimum spanning trees. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1220–1233, 2016. doi:10.1137/1.9781611974331.CH85.

5 Maria-Florina Balcan, Yi Li, David P. Woodruff, and Hongyang Zhang. Testing matrix rank, optimally. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 727–746, 2019. doi:10.1137/1.9781611975482.46.

6 Rémi Bardenet and Odalric-Ambrym Maillard. Concentration inequalities for sampling without replacement. *Bernoulli*, 21(3):1361–1385, 2015. doi:10.3150/14-BEJ605.

7 Tugkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D. Smith, and Patrick White. Testing closeness of discrete distributions. *Journal of the ACM*, 60(1):4:1–4:25, 2013. doi:10.1145/2432622.2432626.

8 Eric Baum and David Haussler. What size net gives valid generalization? *Advances in Neural Information Processing Systems (NIPS)*, 1:81–90, 1988. URL: https://proceedings.neurips.cc/paper/1988/hash/1d7f7abc18fcb43975065399b0d1e48e-Abstract.html.

9 Arnab Bhattacharyya and Yuichi Yoshida. *Property Testing: Problems and Techniques*. Springer Verlag, Berlin, 2022. doi:10.1007/978-981-16-8622-1.

10 Eric Blais. Testing juntas: A brief survey. In Oded Goldreich, editor, *Property Testing: Current Research and Surveys*, pages 32–40. Springer Verlag, Berlin, 2010. doi:10.1007/978-3-642-16367-8_4.

11 Avrim Blum and Ronald Rivest. Training a 3-node neural network is NP-complete. *Advances in Neural Information Processing Systems (NIPS)*, 1:494–501, 1988. URL: https://proceedings.neurips.cc/paper_files/paper/1988/hash/3def184ad8f4755ff269862ea77393dd-Abstract.html.

12 Digvijay Boob, Santanu S. Dey, and Guanghui Lan. Complexity of training ReLU neural network. *Discrete Optimization*, 44:100620, 2022. doi:10.1016/j.disopt.2020.100620.

13 Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. Approximating the minimum spanning tree weight in sublinear time. *SIAM Journal on Computing*, 34(6):1370–1379, 2005. doi:10.1137/S0097539702403244.

14 Sitan Chen, Zehao Dou, Surbhi Goel, Adam R. Klivans, and Raghu Meka. Learning narrow one-hidden-layer ReLU networks. In *Proceedings of the 36th Annual Conference on Learning Theory (COLT)*, pages 5580–5614, 2023. URL: https://proceedings.mlr.press/v195/chen23a.html.

15 Artur Czumaj, Funda Ergün, Lance Fortnow, Avner Magen, Ilan Newman, Ronitt Rubinfeld, and Christian Sohler. Approximating the weight of the Euclidean minimum spanning tree in sublinear time. *SIAM Journal on Computing*, 35(1):91–109, 2005. doi:10.1137/S0097539703435297.

16 Artur Czumaj and Christian Sohler. Sublinear-time algorithms. *Bulletin of EATCS*, 89:23–47, 2006.

17 Artur Czumaj and Christian Sohler. Estimating the weight of metric minimum spanning trees in sublinear time. *SIAM Journal on Computing*, 39(3):904–922, 2009. doi:10.1137/060672121.

18 Artur Czumaj and Christian Sohler. Sublinear-time algorithms. In Oded Goldreich, editor, *Property Testing — Current Research and Surveys*, volume 6390 of *Lecture Notes in Computer Science*, pages 41–64. Springer, 2010. doi:10.1007/978-3-642-16367-8_5.

19 Artur Czumaj and Christian Sohler. Property testing of computational networks. *CoRR arXiv*, abs/2512.07577, 2025. Full version of the current paper. doi:10.48550/arXiv.2512.07577.

20 Hennie Daniels and Marina Velikova. Monotone and partially monotone neural networks. *IEEE Transactions on Neural Networks*, 21(6):906–917, June 2010. doi:10.1109/TNN.2010.2044803.

21 Santanu S Dey, Guanyi Wang, and Yao Xie. Approximation algorithms for training one-node ReLU neural networks. *IEEE Transactions on Signal Processing*, 68:6696–6706, 2020. doi:10.1109/TSP.2020.3039360.

22 Ilias Diakonikolas, Homin K. Lee, Kevin Matulef, Krzysztof Onak, Ronitt Rubinfeld, Rocco A. Servedio, and Andrew Wan. Testing for concise representations. In *Proceedings of the 48th Symposium on Foundations of Computer Science (FOCS)*, pages 549–558, 2007. doi:10.1109/FOCS.2007.32.

23 Gábor Elek. Finite graphs and amenability. *Journal of Functional Analysis*, 263(9):2593–2614, 2012. doi:10.1016/j.jfa.2012.08.021.

24 Eldar Fischer, Guy Kindler, Dana Ron, Shmuel Safra, and Alex Samorodnitsky. Testing juntas. *Journal of Computer and System Sciences*, 68(4):753–787, 2004. doi:10.1016/J.JCSS.2003.11.004.

25 Eldar Fischer, Ilan Newman, and Jirí Sgall. Functions that have read-twice constant width branching programs are not necessarily testable. *Random Structures and Algorithms*, 24(2):175–193, 2004. doi:10.1002/RSA.10110.

26 Vincent Froese, Christoph Hertrich, and Rolf Niedermeier. The computational complexity of ReLU network training parameterized by data dimensionality. *Journal of Artificial Intelligence Research*, 74:1775–1790, 2022. doi:10.1613/jair.1.13547.

27 Surbhi Goel, Adam R. Klivans, Pasin Manurangsi, and Daniel Reichman. Tight hardness results for training depth-2 ReLU networks. In *Proceedings of the 12th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 22:1–22:14, 2021. doi:10.4230/LIPIcs.ITCS.2021.22.

28 Oded Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017. doi:10.1017/9781108135252.

29 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998. doi:10.1145/285055.285060.

30 Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2):302–343, 2002. doi:10.1007/S00453-001-0078-7.

31 Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. doi:10.2307/2282952.

32 Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. doi:10.1016/0893-6080(89)90020-8.

33 Stephen Judd. On the complexity of loading shallow neural networks. *Journal of Complexity*, 4(3):177–192, 1988. doi:10.1016/0885-064X(88)90019-2.

34 Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. doi:10.1038/nature14539.

35 Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. The expressive power of neural networks: A view from the width. In *Proceedings of the 30th Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 6231–6239, 2017. URL: https://papers.nips.cc/paper_files/paper/2017/hash/32cbf687880eb1674a07bf717761dd3a-Abstract.html.

36 Pasin Manurangsi and Daniel Reichman. The computational complexity of training ReLU(s). *CoRR arXiv*, abs/1810.04207, 2018. arXiv:1810.04207.

37 Dan Mikulincer and Daniel Reichman. Size and depth of monotone neural networks: Interpolation and approximation. *IEEE Transactions on Neural Networks and Learning Systems*, 36(4):6314–6325, April 2025. doi:10.1109/TNNLS.2024.3387878.

38 Ilan Newman. Testing membership in languages that have small width branching programs. *SIAM Journal on Computing*, 31(5):1557–1570, 2002. doi:10.1137/S009753970038211X.

39 Ilan Newman and Christian Sohler. Every property of hyperfinite graphs is testable. *SIAM Journal on Computing*, 42(3):1095–1112, 2013. doi:10.1137/120890946.

40 Pekka Orponen. Neural networks and complexity theory. *Nordic Journal of Computing*, 1(1):94–110, 1994. URL: https://www.cs.helsinki.fi/njc/njc1_papers/number1/survey.pdf.

41 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72(6):1012–1042, 2006. doi:10.1016/j.jcss.2006.03.002.

42 Michal Parnas, Dana Ron, and Alex Samorodnitsky. Testing basic Boolean formulae. *SIAM Journal on Discrete Mathematics*, 16(1):20–46, 2002. doi:10.1137/S0895480101407444.

43 Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. OpenAI, 2018. URL: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

44 Dana Ron. Property testing: A learning theory perspective. *Foundations and Trends in Machine Learning*, 1(3):307–402, 2008. doi:10.1561/2200000004.

45 Ronitt Rubinfeld and Asaf Shapira. Sublinear time algorithms. *SIAM Journal on Discrete Mathematics*, 25(4):1562–1588, 2011. doi:10.1137/100791075.

46 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996. doi:10.1137/S0097539793255151.

47 Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014. doi:10.5555/2627435.2670313.

48 Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR arXiv*, abs/1312.6199, 2013. arXiv:1312.6199.

49 Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 30th Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, 2017. URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

50 Heribert Vollmer. *Introduction to Circuit Complexity: A Uniform Approach*. Springer Verlag, Berlin, 1999. doi:10.1007/978-3-662-03927-4.

On the Advantage of Adaptivity for Sampling with Cell Probes

Farzan Byramji
UC San Diego, La Jolla, CA, USA

Daniel M. Kane
UC San Diego, La Jolla, CA, USA

Jackson Morris
UC San Diego, La Jolla, CA, USA

Anthony Ostuni
UC San Diego, La Jolla, CA, USA

Abstract

We construct an explicit distribution $\mathbf{D}$ over $\{0,1\}^N$ that exhibits an essentially optimal separation between adaptive and non-adaptive cell-probe sampling. The distribution can be sampled exactly when each output bit is allowed two adaptive probes to an arbitrarily long sequence of independent uniform symbols from $[N]$. In contrast, any non-adaptive sampler requires $\widetilde{\Omega}(N)$ non-adaptive cell probes to generate a distribution with total variation distance less than $1-o(1)$ from $\mathbf{D}$. This provides a 2-vs-$\widetilde{\Omega}(N)$ separation for sampling with adaptive versus non-adaptive cell probes, improving upon the 2-vs-$\widetilde{\Omega}(\log N)$ separation of Yu and Zhan (ITCS '24) and the $(\log N)^{O(1)}$-vs-$N^{\Omega(1)}$ separation of Alekseev, Göös, Myasnikov, Riazanov, and Sokolov (STOC '26).

2012 ACM Subject Classification Theory of computation → Cell probe models and lower bounds

Keywords and phrases sampling lower bound, cell probe model, adaptive sampling

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.66

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2605.12873

Funding *Farzan Byramji*: Supported by Simons Investigator Award #929894, and NSF Awards CCF-2425349 and AF: Medium 2212136.
Daniel M. Kane: Supported by NSF Medium Award CCF-2107547.
Anthony Ostuni: Partially supported by Simons Investigator Award #929894 and NSF Award CCF-2425349.

Acknowledgements We are grateful to anonymous reviewers for helpful feedback on an earlier version of this manuscript.

1 Introduction

In recent years, the family of computational tasks involving *sampling* has received increased attention, particularly from a complexity-theoretic viewpoint. Broadly, the goal is to take uniform randomness as input and produce samples from some target distribution. Sampling has connections and applications to many other areas of complexity including data structure lower bounds [12, 11, 2, 14, 5, 15, 16, 10, 1], quantum-classical separations [3, 10, 9], randomness extractors [7, 13, 5, 4], and more.

One may study the complexity of sampling tasks in various computational models (e.g., circuits, branching programs, bounded-space Turing machines) and with a variety of randomness sources, but in this work we focus on the *cell-probe* model with uniformly random inputs. In this setting the sampler has query access to arbitrarily many independent


© Farzan Byramji, Daniel M. Kane, Jackson Morris, and Anthony Ostuni;
licensed under Creative Commons License CC-BY 4.0

samples drawn uniformly from $[N] := \{1, 2, \ldots, N\}$, and outputs are determined by arbitrary functions depending on some subset of inputs queried, i.e., our sampler corresponds to some function $f\colon [N]^m \to \{0,1\}^N$, and we consider the distribution on $\{0,1\}^N$ which results from evaluating f on uniformly random input. We write $f(\mathbf{U}_{[N]}^m)$ to denote this distribution where $\mathbf{U}_{[N]}^m$ is the uniform distribution on $[N]^m$. We say that f is d-local if every output bit is a function of at most d input symbols; that is, it can be computed with at most d non-adaptive cell probes.

We also consider an adaptive version of this model wherein the sampler can perform intermediate computations with later queries depending on the results of earlier ones. Indeed, the focus of this work is on understanding the relative power of adaptivity for sampling tasks in this model:

How much can adaptive queries help to sample in the cell-probe model?

The challenge of exhibiting a distribution which requires asymptotically fewer adaptive queries than non-adaptive queries was raised by Viola in [14] and reiterated in [15]. He suggested the following candidate distribution $\mathbf{D}_V$ on $[N]^N$. Fix some integer $k = N^{1-\Omega(1)}$, and sample a string $r \in [N]^k$ uniformly at random. Each output symbol x_i of $\mathbf{D}_V$ is determined by independently and uniformly sampling an index $j \in [k]$ and setting $x_i = r_j$. It is easy to see that two adaptive probes are sufficient to sample from this distribution (assuming $k|N$), but a separation was not known until Yu and Zhan [16] proved that $\Omega(\log N/\log\log N)$ non-adaptive probes are necessary to sample from $\mathbf{D}_V$ (even approximately).

More recently, Alekseev, Göös, Myasnikov, Riazanov, and Sokolov [1] showed that sampling a uniform random permutation in $[N]^N$ also gives a separation between adaptive and non-adaptive cell-probe samplers. They proved that $N^{\Omega(1)}$ non-adaptive probes are required, while Czumaj [6] showed that $O(\log^2 N)$ adaptive probes suffice to get polynomially small distance. Alekseev et al. [1] also showed that $(\log N)^{\Omega(1)}$ adaptive probes are necessary, and highlighted obtaining an $O(1)$-vs-$N^{\Omega(1)}$ separation as an interesting open question. Weaker lower bounds on the number of non-adaptive probes required to sample permutations had been shown earlier in [14, 16].

In this work, we give an essentially optimal $O(1)$-vs-$\widetilde{\Omega}(N)$ separation.

▶ **Theorem 1.** *For any $\varepsilon > 0$ and infinitely many N, there exists an explicit distribution $\mathbf{D}_1$ over $\{0,1\}^N$ which can be sampled exactly by two adaptive probes to independent uniform symbols from $[N]$, yet for any $(N^{1-\varepsilon})$-local function $f\colon [N]^m \to \{0,1\}^N$, we have*

$$\left\| f(\mathbf{U}_{[N]}^m) - \mathbf{D}_1 \right\|_{\mathsf{TV}} \geq 1 - \exp(-N^{\Omega_\varepsilon(1)}). \tag{1}$$

Similarly, there exists a constant $c > 0$ such that for infinitely many N, there exists an explicit distribution $\mathbf{D}_2$ over $\{0,1\}^N$ which can be sampled exactly by two adaptive probes to independent uniform symbols from $[N]$, yet for any $(N/\log^c N)$-local function $g\colon [N]^m \to \{0,1\}^N$, we have

$$\left\| g(\mathbf{U}_{[N]}^m) - \mathbf{D}_2 \right\|_{\mathsf{TV}} \geq 1 - \exp(-\Omega(\log^2 N)). \tag{2}$$

▶ Remark 2. These results are essentially the best possible in several ways.

- Any adaptive sampler which only makes one cell probe (per output bit) is trivially equivalent to a non-adaptive sampler, so at least two probes are necessary for a separation.
- The error in (1) cannot be appreciably improved, since any distribution over $\{0,1\}^N$ can be approximated to distance $1 - e^{-\Omega(N)}$ by a point mass (which needs no locality to generate).

- The locality in (2) cannot be appreciably improved, since a $\Theta(N/\log N)$-local sampler can use the probed cells as a shared seed with $2^{\Theta(N)}$ possible values to approximate any distribution over $\{0,1\}^N$ with exponentially small error (see, e.g., [12, Lemma 5.2]).
- The alphabet size s of the random symbols must be large for such a separation to hold. If s were smaller than any polynomial in N, then any sampler making $d = O(1)$ adaptive probes could be simulated by one making $s^{O(d)} \ll N^{\Omega(1)}$ non-adaptive probes by considering all possible queried inputs.

The distributions in Theorem 1 are instantiations of the following distribution with different choices of parameters.

▶ **Definition 3** (Distribution $\mathbf{D}_{n,\ell}$). *Let $n \geq 2$ be a power of two, and let $\ell \geq 1$ be an integer. We define $\mathbf{D}_{n,\ell}$ to be the distribution over $(\mathbf{i}^{(1)}, \ldots, \mathbf{i}^{(\ell)}, \mathbf{x}, \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(\ell)}) \in \{0,1\}^{\ell\log(n)+(\ell+1)n}$, where each $\mathbf{i}^{(j)}$ is independently uniform over $\{0,1\}^{\log(n)}$ (viewed as an element of $[n]$), $\mathbf{x}$ is uniform over $\{0,1\}^n$, and $\mathbf{y}^{(j)}_k = \mathbf{x}_{k+\mathbf{i}^{(j)} \bmod n}$ for all $j \in [\ell]$ and $k \in [n]$.*

Note that $\mathbf{D}_{n,\ell}$ is similar in spirit to the aforementioned distribution $\mathbf{D}_V$, but with a more explicit indexing structure. In $\mathbf{D}_V$, each output samples a private index $j \in [k]$ and outputs r_j. This produces clusters of coordinates which must take the same value, although the output itself does not reveal which coordinates belong to which cluster. Our method instead explicitly produces many random groups of $\ell + 1$ outputs that must correlate and encodes which clusters those must be in the index portion of the output.

As with $\mathbf{D}_V$, the distribution $\mathbf{D}_{n,\ell}$ can be sampled with two adaptive cell probes.

▷ Claim 4. Let n, ℓ be as in Definition 3, and let N be a (positive) integer multiple of $2n$. Then $\mathbf{D}_{n,\ell}$ can be sampled exactly by two adaptive probes to independent uniform symbols from $[N]$.

Proof. Let the first $\ell + n$ random input symbols be $S_1, \ldots, S_\ell, R_1, \ldots, R_n \in [N]$. For each $j \in [\ell]$, use S_j to generate the shift $\mathbf{i}^{(j)}$ using some balanced mapping from $[N] \to [n]$. Similarly for each $k \in [n]$, use R_k to generate the bit $\mathbf{x}_k$ using some balanced mapping from $[N] \to \{0,1\}$. (Since $(2n)|N$, both mappings exist.) To produce $\mathbf{y}^{(j)}_k$, first probe S_j to learn the shift $\mathbf{i}^{(j)}$, then probe $R_{k+\mathbf{i}^{(j)} \bmod n}$ and output the corresponding bit. ◁

Our main technical result is the following lower bound, which shows that $\mathbf{D}_{n,\ell}$ is hard to sample non-adaptively.

▶ **Theorem 5.** *Let $m \geq d \geq 1$, $\ell \geq 1$, and $n, N \geq 2$ be integers where n is a power of two. Additionally, let $f\colon [N]^m \to \{0,1\}^{\ell\log(n)+(\ell+1)n}$ be a d-local function, and let $\mathbf{D}_{n,\ell}$ be the distribution in Definition 3. Then $f(\mathbf{U}^m_{[N]})$ and $\mathbf{D}_{n,\ell}$ have total variation distance at least*

$$1 - \exp\left(-\Omega\left(\frac{n}{\ell^2 d \log N}\right)\right) - d\left(\frac{\Theta(d\log N)}{n}\right)^{\ell} - \exp\left(d\ell\log(n)\log(N) - \Omega(n)\right).$$

Taking Theorem 5 as a black-box, we can immediately derive Theorem 1.

Proof of Theorem 1 (assuming Theorem 5). Both $\mathbf{D}_1$ and $\mathbf{D}_2$ will be chosen to be padded versions of $\mathbf{D}_{n,\ell}$ (for different values of ℓ), where $n \geq 2$ is a power of 2, $\ell \geq 1$ is an integer, and $N = 4\ell n$. Note that under these conditions, Claim 4 guarantees we can exactly sample $\mathbf{D}_{n,\ell}$ with two adaptive cell probes, so the same must also be true when the distribution is padded with zeros.

In the case of $\mathbf{D}_1$, we set $\delta = 3\varepsilon/4$, $\ell = \lfloor n^{\delta/3} \rfloor$, and $d = \lceil n^{1-\delta} \rceil$. Then Theorem 5 implies that for any d-local $f\colon [N]^m \to \{0,1\}^{\ell \log n + (\ell+1)n}$, we have

$$\left\| f(\mathbf{U}_{[N]}^m) - \mathbf{D}_{n,\ell} \right\|_{\mathsf{TV}} \geq 1 - \exp(-\widetilde{\Omega}(n^{\delta/3})) \geq 1 - \exp(-\Omega(N^{\varepsilon/6}))$$

for large enough n. Since $d \geq N^{1-\varepsilon}$ and $\mathbf{D}_{n,\ell}$ is a projection of $\mathbf{D}_1$, the desired lower bound for $\mathbf{D}_1$ follows. Similarly in the case of $\mathbf{D}_2$, we set $\ell = \lfloor \log^2 n \rfloor$ and $d = \lceil n/\log^7 n \rceil$. Then for any d-local g, we have

$$\left\| g(\mathbf{U}_{[N]}^m) - \mathbf{D}_{n,\ell} \right\|_{\mathsf{TV}} \geq 1 - \exp(-\Omega(\log^2 n)) \geq 1 - \exp(-\Omega(\log^2 N))$$

for large enough n. Using $d \geq N/\log^{10} N$, we obtain the desired lower bound for $\mathbf{D}_2$. ◀

We will prove Theorem 5 in Section 3 after setting up some preliminary material in the subsequent section.

2 Preliminaries

In this section, we review some basic notation, terminology, and standard results. For a positive integer n, we use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. All logarithms given in the paper are base 2. The indicator function is denoted by $\mathbb{1}(\cdot)$.

Asymptotics

We use the standard $\Omega(\cdot), O(\cdot), \Theta(\cdot)$ asymptotic notation to hide universal positive constants. Occasionally, we will use subscripts to indicate an unspecified dependence on a particular parameter (e.g., $\Omega_\varepsilon(1)$). Additionally, we write $\widetilde{\Omega}(\cdot)$ to suppress polylogarithmic factors.

Cell-Probe Samplers

We study sampling in the cell-probe model. A sampler is given access to an arbitrarily long sequence of independent random cells, each uniformly distributed over $[N]$, and produces an output in $\{0,1\}^n$ according to probes made to the random cells. The complexity of the sampler is the maximum number of cells queried in order to produce any single output bit.

A central distinction is whether these probes are adaptive. In an adaptive sampler, the location of a later probe may depend on the values observed in earlier probes. In a non-adaptive sampler, all probe locations for a given output bit are fixed in advance. In the non-adaptive setting, this cost is often called the *locality* of the sampler. Thus, a function $f\colon [N]^m \to \{0,1\}^n$ is said to be d-local if each output bit of f depends on at most d input cells.

Probability

We use bold letters to denote probability distributions and random variables. We reserve $\mathbf{U}_{[n]}$ for the uniform distribution over $[n]$. For an event $\mathcal{E}$, we define $\mathbf{X}(\mathcal{E})$ to be the probability mass assigned to $\mathcal{E}$ by $\mathbf{X}$. For a function f, we use $f(\mathbf{X})$ to denote the output distribution of $f(\mathbf{x})$ on randomly drawn $\mathbf{x} \sim \mathbf{X}$.

Given a distribution $\mathbf{X}$ and positive integer t, we use $\mathbf{X}^t$ to denote the t-fold product distribution $\mathbf{X} \times \cdots \times \mathbf{X}$. If s is a finite set, we write $\mathbf{X}^s$ to emphasize that the coordinates of $\mathbf{X}^{|s|}$ are indexed by s. We refer to $\mathbf{X}$ as a mixture if it can be written as a convex combination

of other distributions. That is, there exist $c_1, \dots, c_k \in [0,1]$ with $\sum_i c_i = 1$ and distributions $\mathbf{X}_1, \dots, \mathbf{X}_k$ such that $\mathbf{X}(\mathcal{E}) = \sum_{i=1}^k c_i \cdot \mathbf{X}_i(\mathcal{E})$ for every event $\mathcal{E}$. Occasionally, we write this more concisely as $\mathbf{X} = \sum_{i=1}^k c_i \mathbf{X}_i$. We measure the similarity of two (discrete) distributions $\mathbf{P}$ and $\mathbf{Q}$ by the *total variation (TV) distance*

$$\|\mathbf{P} - \mathbf{Q}\|_{\mathsf{TV}} = \max_{\text{event } \mathcal{E}} \mathbf{P}(\mathcal{E}) - \mathbf{Q}(\mathcal{E}) = \frac{1}{2} \sum_x |\mathbf{P}(x) - \mathbf{Q}(x)| \,.$$

We say $\mathbf{P}$ is ε-*close* to $\mathbf{Q}$ if $\|\mathbf{P} - \mathbf{Q}\|_{\mathsf{TV}} \le \varepsilon$, and ε-*far* otherwise.

Concentration Inequalities

We will need the following standard concentration inequality.

▶ **Lemma 6** (Chernoff). *Let $\mathbf{X}_1, \dots, \mathbf{X}_n$ be independent indicator random variables with $\mathbf{Pr}[\mathbf{X}_i = 1] \ge p$ for all i. Then for any $\delta \in (0,1)$, we have*

$$\mathbf{Pr}\left[\sum_i \mathbf{X}_i \le (1-\delta)pn\right] \le \exp\left(-\Omega(\delta^2 pn)\right).$$

We will also require a variant of Lemma 6 in which the random variables are only partially independent. There are a plethora of standard results giving concentration under these weaker conditions (e.g., McDiarmid's or Azuma's inequalities). For our purposes, a lesser-known inequality derived from Shearer's lemma (from information theory) by Gavinsky, Lovett, Saks, and Srinivasan [8] appears to provide the best bounds. A collection of random variables $\mathbf{X}_1, \dots, \mathbf{X}_n$ is said to be a *read-k family* if they are functions of independent random variables $\mathbf{Y}_1, \dots, \mathbf{Y}_m$, where each $\mathbf{Y}_j$ influences at most k of the $\mathbf{X}_i$'s. The following is essentially [8, Theorem 1.1].

▶ **Lemma 7** (Read-k Chernoff). *Let $\mathbf{X}_1, \dots, \mathbf{X}_n$ be a read-k family of indicator random variables with $\mathbf{Pr}[\mathbf{X}_i = 1] \ge p$ for all i. Then for any $\delta \in (0,1)$, we have*

$$\mathbf{Pr}\left[\sum_i \mathbf{X}_i \le (1-\delta)pn\right] \le \exp\left(-\Omega\left(\frac{\delta^2 pn}{k}\right)\right).$$

3 Proof of Theorem 5

In this section, we prove Theorem 5, restated below for the reader's convenience.

▶ **Theorem 5.** *Let $m \ge d \ge 1$, $\ell \ge 1$, and $n, N \ge 2$ be integers where n is a power of two. Additionally, let $f\colon [N]^m \to \{0,1\}^{\ell \log(n) + (\ell+1)n}$ be a d-local function, and let $\mathbf{D}_{n,\ell}$ be the distribution in Definition 3. Then $f(\mathbf{U}^m_{[N]})$ and $\mathbf{D}_{n,\ell}$ have total variation distance at least*

$$1 - \exp\left(-\Omega\left(\frac{n}{\ell^2 d \log N}\right)\right) - d\left(\frac{\Theta(d \log N)}{n}\right)^{\ell} - \exp\left(d\ell \log(n) \log(N) - \Omega(n)\right).$$

At a high level, the proof reduces to a setting in which many groups of output bits that are forced to be equal under $\mathbf{D}_{n,\ell}$ do not share a common input cell. For each such group, this absence of a shared input creates a dichotomy: either the sampler produces unequal bits with noticeable probability, immediately violating the defining constraints of $\mathbf{D}_{n,\ell}$, or the entire group is biased toward one fixed constant value. If the latter occurs for many groups, then the sampler places too much mass on outputs where many coordinates simultaneously

take prescribed values, an event that is exponentially unlikely under $\mathbf{D}_{n,\ell}$. Thus in either case the sampler is far from the target distribution. We now proceed to the details.

For clarity, we will write $\mathbf{U}$ for $\mathbf{U}_{[N]}$ and $\mathbf{D}$ for $\mathbf{D}_{n,\ell}$. It will also be convenient to discuss the indices in the output of $f(\mathbf{U}^m)$ in a similar fashion to the definition of $\mathbf{D}$ (in Definition 3). That is, we will abuse notation and often use (say) $\mathbf{i}^{(1)}$ for the first $\log n$ bits of $f(\mathbf{U}^m)$. It will be evident from context which distribution (i.e., $f(\mathbf{U}^m)$ or $\mathbf{D}$) the random variables $\mathbf{i}^{(j)}, \mathbf{x}, \mathbf{y}^{(j)}$ are being taken from.

Let $t > 0$ be a threshold parameter (eventually taken to be $\Theta(\ell d \log N)$), and let $S \subseteq [m]$ be the set of input symbols to the sampler f that either affect more than t output bits or affect one of the first $\ell \log n$ outputs (corresponding to $\mathbf{i}^{(1)}, \ldots, \mathbf{i}^{(\ell)}$). Observe that $s := |S| \leq d\ell \log(n) + d(\ell+1)n/t$. By conditioning on the values that the symbols in S take, we can express $f(\mathbf{U}^m)$ as the mixture

$$f(\mathbf{U}^m) = \sum_{\rho \in [N]^S} \frac{1}{N^s} \cdot f(\mathbf{U}^{[m]\setminus S}, \rho).$$

Below, we will write the distribution $f(\mathbf{U}^{[m]\setminus S}, \rho)$ more concisely as $\mathbf{F}_\rho$. Observe that any such conditioning ρ fixes the value of the shift vector $\mathbf{i}(\rho) := (\mathbf{i}^{(1)}(\rho), \ldots, \mathbf{i}^{(\ell)}(\rho))$, where $\mathbf{i}^{(j)}(\rho)$ is the value of $\mathbf{i}^{(j)}$ after conditioning on ρ, and it guarantees that every unfixed input cell affects at most t output bits. We will use these properties to show that for certain restrictions ρ, the distribution $\mathbf{F}_\rho$ is far from $\mathbf{D}$ conditioned on their shift vectors being equal. Then, we amalgamate the behavior of these conditional distributions to obtain our final distance bound.

In order for $f(\mathbf{U}^m)$ to approximate $\mathbf{D}$, there must be clusters of $k := \ell + 1$ output bits which all take the same value. More precisely, for each shift vector $i \in (\{0,1\}^{\log n})^\ell$ and $u \in [n]$, define the equality block $B_u(i) = (\mathbf{x}_u, \mathbf{y}^{(1)}_{u-i^{(1)}}, \ldots, \mathbf{y}^{(\ell)}_{u-i^{(\ell)}})$. Call such an i *bad* if at least $n/4$ of the equality blocks have some unfixed input cell $c \notin S$ which affects every bit in the block (and *good* otherwise); if a block has such an input symbol, we say the block is *covered*.

▷ Claim 8. The fraction of bad $i \in (\{0,1\}^{\log n})^\ell$ is at most $4d\left(\frac{t}{\ell n}\right)^\ell$.

Proof. For each unfixed input cell $c \notin S$, define $A_{x,c} \subseteq [n]$ to be the set of $\mathbf{x}$ coordinates affected by c. Similarly, for all $j \in [\ell]$ define $A_{j,c} \subseteq [n]$ to be the set of $\mathbf{y}^{(j)}$ coordinates affected by c. Observe that $\sum_j |A_{j,c}| \leq t$, and that a block $B_u(i)$ is covered by a cell c only if $u \in A_{x,c}$ and $u - i^{(j)} \in A_{j,c}$ for all j. In a uniformly sampled $\mathbf{i}$, each $\mathbf{i}^{(j)}$ is chosen independently and uniformly at random from $\{0,1\}^{\log n}$, so the probability that $u - \mathbf{i}^{(j)} \in A_{j,c}$ is $|A_{j,c}|/n$. Thus,

$$\begin{aligned}
\mathop{\mathbb{E}}_{\mathbf{i}\in(\{0,1\}^{\log n})^\ell}[\#\text{ blocks covered}] &\leq \sum_{c\notin S} \mathop{\mathbb{E}}_{\mathbf{i}\in(\{0,1\}^{\log n})^\ell}[\#\text{ blocks covered by } c] \\
&= \sum_{c\notin S} \sum_{u\in A_{x,c}} \prod_{j=1}^{\ell} \frac{|A_{j,c}|}{n} && \text{(since } \mathbf{i}^{(j)}\text{'s independent)} \\
&\leq \sum_{c\notin S} |A_{x,c}| \left(\frac{\sum_j |A_{j,c}|}{n\ell}\right)^\ell && \text{(by AM-GM inequality)} \\
&\leq \left(\sum_{c\notin S} |A_{x,c}|\right) \cdot \left(\frac{t}{n\ell}\right)^\ell && \text{(by def'n of } S\text{)} \\
&\leq dn \left(\frac{t}{n\ell}\right)^\ell. && \text{(since } \mathbf{x} \text{ affected by} \leq dn \text{ inputs)}
\end{aligned}$$

Applying Markov's inequality concludes the proof. ◁

By Claim 8, we can primarily focus our attention on the case of good i's, where we have many uncovered blocks. We will show that the lack of correlation within the bits of such a block causes behavior which does not occur in the target distribution. Below, we view $\mathbf{Z}$ as the marginal distribution of $\mathbf{F}_\rho$ onto an uncovered block.

▷ Claim 9. Let $\mathbf{Z}$ be the marginal distribution over some $k \geq 2$ output bits. If no input cell affects every bit in $\mathbf{Z}$, then either

1. $\mathbf{Pr}[\mathbf{Z} \notin \{0^k, 1^k\}] \geq 1/(100k)$, or
2. $\mathbf{Pr}[\mathbf{Z} = z] \geq 2/3$ for some fixed $z \in \{0^k, 1^k\}$.

Proof. Arbitrarily partition the input cells into sets $I_1, \ldots, I_k$ where the cells in I_j do not affect the j-th output bit of $\mathbf{Z}$. We proceed by a hybrid argument. Consider two independent random inputs $\mathbf{A}, \mathbf{B}$. We define the sequence of inputs

$$\mathbf{A} = \mathbf{C}^{(0)}, \mathbf{C}^{(1)}, \ldots, \mathbf{C}^{(k)} = \mathbf{B},$$

where $\mathbf{C}^{(j)}$ has the input symbols in $I_1 \cup \cdots \cup I_j$ taken from $\mathbf{B}$ and the rest from $\mathbf{A}$. We also define $\mathbf{Z}^{(j)}$ to be the value of $\mathbf{Z}$ on input $\mathbf{C}^{(j)}$, and observe that each $\mathbf{Z}^{(j)}$ has the same marginal distribution as $\mathbf{Z}$. Moreover, let $\delta = \mathbf{Pr}[\mathbf{Z} \notin \{0^k, 1^k\}]$, $p_0 = \mathbf{Pr}[\mathbf{Z} = 0^k]$, and $p_1 = \mathbf{Pr}[\mathbf{Z} = 1^k]$.

Whenever $\mathbf{Z}^{(0)} = 0^k$ and $\mathbf{Z}^{(k)} = 1^k$ (or vice-versa), there must exist some $\mathbf{Z}^{(j)} \notin \{0^k, 1^k\}$. Indeed, any two consecutive outputs $\mathbf{Z}^{(j-1)}, \mathbf{Z}^{(j)}$ agree on their j-th bit, since the cells in I_j do not affect it. Thus, we cannot transition directly from 0^k to 1^k, so if the sequence contains both strings, then it must also contain some other output. This implies $2p_0p_1 \leq (k+1)\delta$. Assume by contradiction that $\delta < 1/(100k)$ and $p_0, p_1 < 2/3$. Then $\min(p_0, p_1) > 1 - \frac{1}{100k} - \frac{2}{3} \geq \frac{1}{4}$, so

$$\frac{1}{8} \leq 2\left(\frac{1}{4}\right)^2 < \frac{k+1}{100k} \leq \frac{1}{100} + \frac{1}{100k} \leq \frac{1}{8} - \frac{11}{100},$$

a contradiction. ◁

Fix a conditioning ρ such that $\mathbf{i}(\rho)$ is good. That is, at least $3n/4$ blocks $B_u(\mathbf{i}(\rho))$ are uncovered. We break into cases depending on which conclusion of Claim 9 is satisfied by most of these blocks. Below, let $n' = 3n/8$.

Case 1: Most blocks satisfy (1)

Let $B_1, \ldots, B_{n'}$ be blocks satisfying (1). While they are not fully independent, one would expect their behavior to resemble that of independent random variables, since each unfixed input cell only affects a small number of blocks.[1] More formally, we have that the indicator random variables $\{\mathbb{1}(B_j \notin \{0^k, 1^k\})\}_j$ form a read-t family, so Lemma 7 implies

$$\mathop{\mathbf{Pr}}_{\mathbf{F}_\rho}\left[\sum_{j=1}^{n'} \mathbb{1}(B_j \notin \{0^k, 1^k\}) \geq \frac{n'}{200k}\right] \geq 1 - \exp\left(-\Omega\left(\frac{n}{kt}\right)\right).$$

For comparison, this event never occurs under $\mathbf{D}$ conditioned on its shift vector being $\mathbf{i}(\rho)$.

[1] A greedy construction allows us to find $\Omega(n/dt)$ actually independent blocks, but pursuing this line of analysis gives worse bounds.

Case 2: Most blocks satisfy (2)

Let $B_1, \dots, B_{n'}$ be blocks satisfying (2), where their most common values are $b_1, \dots, b_{n'}$, respectively. Again applying Lemma 7, we have

$$\Pr_{\mathbf{F}_\rho}\left[\sum_{j=1}^{n'} \mathbb{1}(B_j = b_j) \geq \frac{7n'}{12}\right] \geq 1 - \exp\left(-\Omega\left(\frac{n}{t}\right)\right).$$

For comparison, $\sum_{j=1}^{n'} \mathbb{1}(B_j = b_j)$ is distributed like the binomial $\mathsf{Bin}(n', 1/2)$ under $\mathbf{D}$ conditioned on its shift vector being $\mathbf{i}(\rho)$, so this event occurs with probability at most $e^{-\Omega(n)}$ by Lemma 6.

In either case, there exists an event $\mathcal{E}_\rho$ which occurs with probability at least $1 - e^{-\Omega\left(\frac{n}{kt}\right)}$ under $\mathbf{F}_\rho$, but probability at most $e^{-\Omega(n)}$ under $\mathbf{D}$ conditioned on its shift vector. We now define the global event $\mathcal{E}$ witnessing the TV distance between $f(\mathbf{U}^m)$ and $\mathbf{D}$ to be

$$\mathcal{E} = \{\text{bad } \mathbf{i}\} \cup \bigcup_{\rho\,:\,\mathbf{i}(\rho) \text{ is good}} (\mathbf{i} = \mathbf{i}(\rho) \text{ and } \mathcal{E}_\rho).$$

Under $f(\mathbf{U}^m)$, each conditioning $\rho \in [N]^S$ produces a shift vector $\mathbf{i}(\rho)$. If $\mathbf{i}(\rho)$ is bad, then the output automatically lies in $\mathcal{E}$, and if $\mathbf{i}(\rho)$ is good, then we have already shown it lies in $\mathcal{E}$ with probability at least $1 - e^{-\Omega\left(\frac{n}{kt}\right)}$. To analyze the probability under $\mathbf{D}$, we apply a union bound to find that

$$\begin{aligned}\Pr_{\mathbf{D}}[\mathcal{E}] &\leq \Pr_{\mathbf{D}}[\mathbf{i} \text{ is bad}] + \sum_{\rho\,:\,\mathbf{i}(\rho) \text{ is good}} \Pr_{\mathbf{D}}\left[\mathcal{E}_\rho \,|\, \mathbf{i} = \mathbf{i}(\rho)\right] \\ &\leq 4d\left(\frac{t}{\ell n}\right)^\ell + N^s \exp(-\Omega(n)). \qquad \text{(by Claim 8)}\end{aligned}$$

Hence,

$$\|f(\mathbf{U}^m) - \mathbf{D}\|_{\mathsf{TV}} \geq 1 - \exp\left(-\Omega\left(\frac{n}{(\ell+1)t}\right)\right) - 4d\left(\frac{t}{\ell n}\right)^\ell - N^s \exp(-\Omega(n)).$$

Setting $t = \Theta(\ell d \log N)$ with a sufficiently large implicit constant gives $s \leq d\ell \log(n) + O(n/\log N)$, where $O(n/\log N)$ has a sufficiently small implicit constant, so we find the distance between $f(\mathbf{U}^m)$ and $\mathbf{D}$ is at least

$$1 - \exp\left(-\Omega\left(\frac{n}{\ell^2 d \log N}\right)\right) - d\left(\frac{\Theta(d \log N)}{n}\right)^\ell - \exp\left(d\ell \log(n) \log(N) - \Omega(n)\right).$$

This concludes the proof of Theorem 5.

References

1 Yaroslav Alekseev, Mika Göös, Konstantin Myasnikov, Artur Riazanov, and Dmitry Sokolov. Sampling permutations with cell probes is hard. In *Proceedings of the 58th Annual ACM Symposium on Theory of Computing*, 2026.

2 Chris Beck, Russell Impagliazzo, and Shachar Lovett. Large deviation bounds for decision trees and sampling lower bounds for AC0-circuits. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 101–110. IEEE, 2012. doi:10.1109/FOCS.2012.82.

3 Adam Bene Watts and Natalie Parham. Unconditional quantum advantage for sampling with shallow circuits. In *17th Innovations in Theoretical Computer Science Conference*, volume 362 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 17, 12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. doi:10.4230/lipics.itcs.2026.17.

4 Farzan Byramji, Daniel M Kane, Jackson Morris, and Anthony Ostuni. Hard-to-sample distributions from robust extractors. *arXiv preprint arXiv:2604.26179*, 2026. doi:10.48550/arXiv.2604.26179.

5 Eshan Chattopadhyay, Jesse Goodman, and David Zuckerman. The space complexity of sampling. In *13th Innovations in Theoretical Computer Science Conference (ITCS)*, 2022. doi:10.4230/LIPIcs.ITCS.2022.40.

6 Artur Czumaj. Random permutations using switching networks. In *STOC'15—Proceedings of the 2015 ACM Symposium on Theory of Computing*, pages 703–712. ACM, New York, 2015. doi:10.1145/2746539.2746629.

7 Anindya De and Thomas Watson. Extractors and lower bounds for locally samplable sources. *ACM Transactions on Computation Theory (TOCT)*, 4(1):1–21, 2012. doi:10.1145/2141938.2141941.

8 Dmitry Gavinsky, Shachar Lovett, Michael Saks, and Srikanth Srinivasan. A tail bound for read-k families of functions. *Random Structures Algorithms*, 47(1):99–108, 2015. doi:10.1002/rsa.20532.

9 Daniel Grier, Daniel M. Kane, Jackson Morris, Anthony Ostuni, and Kewen Wu. Quantum advantage from sampling shallow circuits: beyond hardness of marginals. In *17th Innovations in Theoretical Computer Science Conference*, volume 362 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 73, 14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. doi:10.4230/lipics.itcs.2026.73.

10 Daniel M Kane, Anthony Ostuni, and Kewen Wu. Locality bounds for sampling Hamming slices. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1279–1286, 2024. doi:10.1145/3618260.3649670.

11 Shachar Lovett and Emanuele Viola. Bounded-depth circuits cannot sample good codes. In *2011 IEEE 26th Annual Conference on Computational Complexity*, pages 243–251. IEEE, 2011. doi:10.1109/CCC.2011.11.

12 Emanuele Viola. The complexity of distributions. *SIAM Journal on Computing*, 41(1):191–218, 2012. doi:10.1137/100814998.

13 Emanuele Viola. Extractors for circuit sources. *SIAM Journal on Computing*, 43(2):655–672, 2014. doi:10.1137/11085983X.

14 Emanuele Viola. Sampling lower bounds: boolean average-case and permutations. *SIAM Journal on Computing*, 49(1):119–137, 2020. doi:10.1137/18M1198405.

15 Emanuele Viola. New Sampling Lower Bounds via the Separator. In *38th Computational Complexity Conference (CCC 2023)*, volume 264 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 26:1–26:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.CCC.2023.26.

16 Huacheng Yu and Wei Zhan. Sampling, Flowers and Communication. In *15th Innovations in Theoretical Computer Science Conference (ITCS 2024)*, volume 287 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 100:1–100:11. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ITCS.2024.100.

Fast List Recovery of Univariate Multiplicity Codes

Rohan Goyal
Massachusetts Institute of Technology, Cambridge, MA, USA

Prahladh Harsha
Tata Institute of Fundamental Research, Mumbai, India

Mrinal Kumar
Tata Institute of Fundamental Research, Mumbai, India

Ashutosh Shankar
Tata Institute of Fundamental Research, Mumbai, India

Abstract

Recent work gave near-linear time algorithms for list decoding Folded Reed–Solomon codes and univariate multiplicity codes up to capacity in their natural parameter regimes. Unlike most known list decoding algorithms, these techniques appeared inherently tied to list decoding, and it was unclear whether they could be extended to list recovery in near-linear time.

In this work, we resolve this question by giving $\widetilde{O}(n)$-time algorithms for list recovery of Folded Reed–Solomon codes and univariate multiplicity codes up to capacity, where n is the block length. Our algorithms build on the lattice-based framework of the prior work, augmented with a new technical ingredient: the construction of suitably structured lattices over the univariate polynomial ring that capture the list recovery problem for these codes.

2012 ACM Subject Classification Theory of computation → Error-correcting codes; Theory of computation → Design and analysis of algorithms

Keywords and phrases list-recovery, multiplicity-codes, near-linear-algorithm

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.67

Category RANDOM

Related Version *Full Version*: arXiv:2512.00248

Funding *Rohan Goyal*: Supported by (Yael Tauman Kalai's) grant from Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-25-C-0300. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA).
Prahladh Harsha: Supported by the Department of Atomic Energy, Government of India, under project number RTI400112 and in part by a Google research grant.
Mrinal Kumar: Supported by the Department of Atomic Energy, Government of India, under project number RTI400112, and in part by research grants from Google, SERB and Premji Invest.
Ashutosh Shankar: Supported by the Department of Atomic Energy, Government of India, under project number RTI400112.

1 Introduction

An error-correcting code $\mathcal{C} \subseteq \Sigma^n$ is said to be (η, L) list decodable if every Hamming ball of radius ηn in Σ^n contains at most L codewords of $\mathcal{C}$. When $\eta < \delta/2$, where δ is the minimum distance of the code, we are in the unique decoding regime, in which at most one codeword lies in such a ball. List decoding is of fundamental interest both from a combinatorial perspective, where one studies tradeoffs between parameters such as η, L, δ, and the rate of the code, and from an algorithmic perspective, where the goal is to design efficient decoding algorithms for increasingly large values of η. Since Sudan's breakthrough work showing that Reed–Solomon

© Rohan Goyal, Prahladh Harsha, Mrinal Kumar, and Ashutosh Shankar;
licensed under Creative Commons License CC-BY 4.0

codes can be efficiently list decoded beyond the unique decoding radius [19], there has been substantial progress in our understanding of list decoding, culminating in explicit codes that achieve list decoding capacity with constant list size [9, 8, 10, 14, 15, 17, 5, 11] and, in several cases, near-linear time decoding algorithms [6, 2, 18]. Beyond coding theory, list decoding has deep and fruitful connections to complexity theory and pseudorandomness, including randomness extraction and hardness amplification [21].

Our focus in this paper is on the related notion of *list recovery*. A code $\mathcal{C} \subseteq \Sigma^n$ is said to be (η, ℓ, L) list recoverable if for all subsets $E_1, \ldots, E_n \subseteq \Sigma$ with $|E_i| \leq \ell$, there are at most L codewords $c \in \mathcal{C}$ such that $c_i \notin E_i$ for at most ηn coordinates $i \in [n]$. Clearly, the case $\ell = 1$ recovers the definition of list decoding. List recovery has attracted considerable attention both because of its close connection to list decoding and because of its numerous applications, including the construction of pseudorandom objects such as extractors and expanders, cryptographic constructions, and algorithmic problems such as group testing and compressed sensing. We refer the reader to the survey of Resch and Venkitesh [16] and the lecture notes of Wootters [23] for further background.

A notable feature of many list decoding algorithms is that they naturally extend to the more general setting of list recovery. This phenomenon holds for the classical Sudan and Guruswami–Sudan algorithms for Reed–Solomon codes [19, 9], for algorithms achieving capacity for Folded Reed–Solomon (FRS) and (univariate and multivariate) multiplicity codes [8, 10, 14, 15, 3, 4], as well as for recent results on expander-based codes approaching list decoding capacity [11, 18, 12]. Consequently, one might expect fast list decoding algorithms to extend automatically to fast list recovery.

A central direction in the study of list decoding and list recovery is the design of near-linear time algorithms. This includes both the construction of codes admitting such fast decoding algorithms and the development of efficient algorithms for several well-studied and natural code families. A key milestone in this direction is the work of Alekhnovich [1], which gave a near-linear time algorithm for list decoding (and list recovery) Reed–Solomon codes up to the Johnson radius. Alekhnovich's algorithm can be viewed as a fast implementation of the Guruswami–Sudan algorithm that exploits natural connections between polynomial interpolation-based decoding algorithms and lattices over the univariate polynomial ring. More recently, a lattice-based framework building on these ideas was used to obtain near-linear time list decoding algorithms up to capacity for FRS codes and univariate multiplicity codes [6]. In addition, this work developed fast algorithms for solving certain families of differential and functional equations that arise naturally in this context.

Perhaps surprisingly, the techniques underlying these near-linear time list decoding results do not appear to extend directly to the setting of list recovery. This obstruction already arises for $\ell = 2$, and stands in contrast to the behavior of most known list decoding algorithms. In this work, we show that this limitation can in fact be overcome[1].

Informally, our main result gives a near-linear time algorithm for list recovery up to capacity. In the standard parameter regime for univariate multiplicity codes, where the list-size parameter ℓ, the error parameter ε, and the multiplicity parameter s are treated as constants, the running time of our algorithm is $\widetilde{O}(n)$.

More formally, we prove the following theorem.

▶ **Theorem 1** (main result for multiplicity codes). *For every $\varepsilon > 0$ and $\ell \in \mathbb{N}$, there exists $s_0 = \Theta(\ell/\varepsilon^2) \in \mathbb{N}$ such that for all $s > s_0$, degree parameter k, block length n, and fields $\mathbb{F}$ of characteristic zero or characteristic greater than k, satisfying $k < ns$, the following holds.*

[1] In this paper we focus on univariate multiplicity codes. We believe our results extend to FRS codes and defer that case to the full version.

There is a randomized algorithm that, given a set $S \subseteq \mathbb{F}$ of size n and sets $E_\alpha \subseteq \mathbb{F}^s$ with $|E_\alpha| \le \ell$ for each $\alpha \in S$, takes

$$O\Big(n \cdot \mathsf{poly}\Big(s, \ell, \log n, (\ell/\varepsilon)^{\frac{1+\log \ell}{\varepsilon}}\Big)\Big)$$

field operations and with high probability outputs all univariate polynomials $f(X) \in \mathbb{F}[X]$ of degree less than k such that for at least $(\frac{k}{sn} + \varepsilon)$ fraction of $\alpha \in S$,

$$(f(\alpha), f^{(1)}(\alpha), \ldots, f^{(s-1)}(\alpha)) \in E_\alpha.$$

1.1 An overview of the proof

We give a high-level overview of the proof, focusing on the list-recovery problem for univariate multiplicity codes. Our algorithm follows the same two-step structure as prior near-linear time list-decoding algorithms, but the interpolation step requires a fundamentally new construction.

The input to the list-recovery problem consists of a set $S \subseteq \mathbb{F}$ of size n, where each $\alpha \in S$ is associated with a list $\{\beta(\alpha)_j\}_{j\in[\ell]}$ of s-tuples in $\mathbb{F}^s$. As in the list-decoding setting, we refer to this input as the received word.

We begin by recalling the structure of the near-linear time list-decoding algorithm of [6], which is a fast implementation of the algorithm of Guruswami and Wang [10]. This algorithm has two main components:

- **Interpolation via a differential equation.** One constructs, for an appropriately chosen parameter $m = \Theta(\sqrt{s})$, a nonzero polynomial $Q(X, Y_0, \ldots, Y_m) = \widetilde{Q}(X) + \sum_i Q_i(X)Y_i$, linear in the Y-variables and of bounded degree, such that every polynomial f whose encoding agrees sufficiently with the received word satisfies the differential equation:

  $$Q(X, f(X), f^{(1)}(X), \ldots, f^{(m)}(X)) \equiv 0.$$

- **Solving the differential equation.** The solution space of this linear differential equation is an affine space of dimension at most m. A near-linear time algorithm computes a basis for this space, followed by a pruning step [15, 20] to recover the constant-sized list of close enough codewords.

The fast list-recovery algorithm in this paper follows the same high-level outline. This is consistent with the polynomial-time list-recovery algorithm of Guruswami and Wang [10], where the list-decoding and list-recovery procedures are structurally identical, differing only in parameter settings.

To extend the results of [6] to the list-recovery setting, we use the fast differential equation solver from [6] as is. The main new ingredient is in the first step: constructing an appropriate differential equation in near-linear time.

We briefly recall the relevant interpolation framework from [6]. In the list-decoding setting, each $\alpha \in S$ is associated with a unique received vector $\beta(\alpha)$. The algorithm constructs in near-linear time univariate polynomials $A_0(X), A_1(X), \ldots, A_m(X)$ satisfying $(A_i)^{(j)}(\alpha) = \beta(\alpha)^{(i+j)}$ for all $\alpha \in S$, $i \in [m], j \in [s-m]$. One then considers the $\mathbb{F}[X]$-module (lattice) of polynomials in $X, Y_0, \ldots, Y_m$ generated by

$$\{\, Y_i - A_i(X) : i \in [m] \,\} \;\cup\; \left\{ \prod_{\alpha \in S} (X - \alpha)^{s-m} \right\}. \tag{1}$$

This lattice has the property that any nonzero element of this lattice of sufficiently small degree yields a polynomial $Q(X, Y_0, \ldots, Y_m) = \widetilde{Q}(X) + \sum_i Q_i(X)Y_i$ such that every polynomial f whose encoding agrees with the received word on sufficiently many α's satisfies

$$Q(X, f, f^{(1)}, \ldots, f^{(m)}) \equiv 0.$$

In [6], existence of such a "short" nonzero lattice element is proved via an analogue of Minkowski's theorem, and it is found in near-linear time using Alekhnovich's algorithm for shortest vectors (with respect to the degree norm) in polynomial lattices. The "short" guarantee here matches that [10] achieve via variable and constraint counting.

The difficulty in moving to list recovery is that each α now comes with ℓ candidate vectors $\beta(\alpha)_1, \ldots, \beta(\alpha)_\ell$, so the received word is no longer a function $\alpha \mapsto \beta(\alpha)$. In particular, the definition of the polynomials A_i above breaks down. A natural attempt is to arbitrarily "split" the input into ℓ received words in the list-decoding sense: for $t \in [\ell]$, let $r_t = (\alpha, \beta(\alpha)_t)_{\alpha \in S}$, and construct the corresponding polynomials $A_{t,0}, \ldots, A_{t,m}$. One could then try to work with the lattice generated by products such as $\prod_{t=1}^{\ell}(Y_j - A_{t,j}(X))$ (over $\mathbb{F}[X]$), thereby capturing the disjunction "Y_j matches one of the ℓ possibilities." However, this immediately leads to polynomials Q with Y-degree ℓ, and hence to nonlinear differential equations. This is the main obstacle. Even if one could find such a low-degree Q in near-linear time, it is unclear how to solve the resulting nonlinear differential equation and recover all close codewords efficiently: the solution set need not be an affine subspace, and the linear-algebraic techniques of [6] do not apply. While high Y-degree interpolation is standard in other contexts (e.g., [19, 9, 1, 14, 13]), we do not know how to make this approach work here[2].

Motivated by this, it is natural to insist on keeping Q *linear* in the Y-variables even in the list-recovery setting. Guruswami and Wang show that this can be done in polynomial time. Our main technical contribution is to show that it can in fact be done in near-linear time: we construct a carefully structured lattice over $\mathbb{F}[X]$ whose generators are linear in $Y_0, \ldots, Y_m$, and whose short nonzero vectors correspond precisely to linear differential equations that *capture* all close codewords. In the list-decoding setting the received word is a function $\alpha \mapsto \beta(\alpha)$ leading to an almost-diagonal basis of the form (1) for the resulting $(m+1)$-dimensional lattice. In list recovery, the received word no longer has such a functional structure and hence one cannot expect the resulting lattice to admit a similar simple basis. What we show is the next best thing: there exists a lattice whose basis has a structured almost-triangular form, which is sufficient for computing short nonzero vectors efficiently and for deriving the required linear differential equation. The basis we obtain has the following form.

$$\begin{aligned}
&\prod_{\alpha \in S}(X - \alpha)^{s-m}, && \\
&Y_i \cdot \prod_{\alpha \in S}(X - \alpha)^{s-m}, && \forall i, \text{ such that } 0 \le i \le \ell - 2 \\
&\widetilde{C_i}(X) + \sum_{r=0}^{i-1} Y_r \cdot C_{i,r}(X) + Y_i \cdot \prod_{\alpha \in S}(X - \alpha)^{\ell-1}, && \forall i, \text{ such that } \ell - 1 \le i \le m
\end{aligned}$$

for some polynomials $\widetilde{C_i}, C_{i,r} \in \mathbb{F}[X]$ for $\ell - 1 \le i \le m$ and $0 \le r \le i - 1$. The proof that such a basis exists is considerably involved. We first show that such an almost-triangular

[2] Note that the Guruswami–Sudan list-decoding algorithm, and Alekhnovich's near-linear time implementation thereof, already employ a Y-degree larger than one; consequently, Alekhnovich's approach readily extends to list recovery.

basis exists for each $\alpha \in S$ and then combine the almost-triangular bases for the different α to obtain the basis of the above type using the Chinese remainder theorem. See Section 3 for details.

Organization

The rest of this paper is organized as follows. Section 2 introduces univariate multiplicity codes and the subroutines used in the lattice construction. Section 3 gives the fast lattice construction and the resulting differential equation. Setting up this lattice is the key novelty of the paper. Section 4 describes the list-extraction procedure from the differential equation, completing the proof of Theorem 1.

2 Preliminaries

2.1 Notation

- We work in some fixed finite field $\mathbb{F}$.
- For a natural number N, we will use $[N]$ to refer to the set $\{0, 1, 2, \ldots, N-1\}$.
- We use $\mathbf{0}_k$ to denote the k-length zero vector. We may drop the subscript if the length is clear from context.

2.2 Univariate multiplicity codes

We recall the formal definition of the univariate multiplicity code.

▶ **Definition 2** (Multiplicity codes). *Let n, k, s be positive integers satisfying $k < ns$, $\mathbb{F}$ be a field of characteristic zero or at least s and size at least n, and let $S = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ be a set of n distinct elements of $\mathbb{F}$. Then, the univariate multiplicity code with multiplicity parameter s and degree parameter k is defined as follows.*

The alphabet of the code is $E = \mathbb{F}^s$ and the block length is $n = |S|$, where the coordinates of a codeword are indexed by $\alpha \in S$. The message space is the set of all univariate polynomials of degree less than k in $\mathbb{F}[x]$. The encoding of such a message $f \in \mathbb{F}[x]$, denoted by $\mathsf{Mult}_s(f) \in E^S$ is defined as

$$\mathsf{Mult}_s(f)|_\alpha := \left(f(\alpha), f^{(1)}(\alpha), \ldots, f^{(s-1)}(\alpha)\right),$$

where $f^{(j)}(\alpha)$ is the evaluation of the jth-order derivative[3] of f on the input α. The rate of this code is k/ns while the distance is $1 - (k-1)/ns$.

In the list-recovery setting, the received word R assigns to every evaluation point $\alpha \in S$ a collection of tuples $\beta(\alpha)_j = (\beta(\alpha)_j^{(0)}, \beta(\alpha)_j^{(1)}, \ldots, \beta(\alpha)_j^{(s-1)})$ for every $j \in [\ell]$. In words $\beta(\alpha)_j^{(i)}$ is the jth option for the ith derivative of the message polynomial at α. Agreement is with respect to inclusion in this list; that is, $\mathsf{Mult}(f)$ agrees with R on point α if there exists $j \in [\ell]$ such that $(f(\alpha), f^{(1)}(\alpha), \ldots, f^{(s-1)}(\alpha)) = \beta(\alpha)_j = (\beta(\alpha)_j^{(0)}, \beta(\alpha)_j^{(1)}, \ldots, \beta(\alpha)_j^{(s-1)})$.

Our goal is to carry out list recovery from an agreement fraction of $k/ns + \varepsilon$, for a given parameter ε.

[3] This definition in terms of (standard) derivatives requires the characteristic of the field to be 0 or at least k. All known capacity-achieving list-decoding results for univariate multiplicity codes require large characteristic (previous works [14, 10, 15, 6] as well as our work). For this reason as well as ease of presentation, we work with standard derivatives.

2.3 Polynomial operators and lattices

We recall below the τ operator, used in [10, 13], and [6].

▶ **Definition 3** (Tau operator). *Let $s \in \mathbb{N}$ be a parameter. Then, for every m such that $0 \leq m < s$, the function τ is an $\mathbb{F}$-linear map from the set of polynomials in $\mathbb{F}[X, Y_0, Y_1, \ldots, Y_m]$ with $\mathbf{Y}$-degree 1 to the set of polynomials in $\mathbb{F}[X, Y_0, Y_1, \ldots, Y_{m+1}]$ with $\mathbf{Y}$-degree 1 and is defined as follows.*

$$\tau\left(\widetilde{Q}(X) + \sum_{i=0}^{m} Q_i(X) \cdot Y_i\right) := \widetilde{Q}^{(1)}(X) + \sum_{i=0}^{m} \left(Q_i^{(1)}(X) \cdot Y_i + Q_i(X) \cdot Y_{i+1}\right).$$

For an integer $i \leq (s-m-1)$, we use $\tau^{(i)}$ to denote the linear map from $\mathbb{F}[X, Y_0, Y_1, \ldots, Y_m]$ to $\mathbb{F}[X, Y_0, Y_1, \ldots, Y_{m+i}]$ obtained by applying the operator τ iteratively i times.

The operator is defined this way so that

$$\frac{d}{dX} Q(X, f(X), f^{(1)}(X), \ldots, f^{(m)}(X)) = \tau Q(X, f(X), f^{(1)}(X), \ldots, f^{(m+1)})(X).$$

It will be convenient to view polynomials in $\mathbb{F}[X, Y_0, \ldots, Y_{m+i}]$ for $i \leq s-m-1$ as polynomials in $\mathbb{F}[X, Y_0, \ldots, Y_{s-1}]$. Thus, when evaluating a polynomial $R \in \mathbb{F}[X, Y_0, \ldots, Y_{m+i-1}]$ at the point $(\alpha, \beta) \in \mathbb{F}^{s+1}$, $R(\alpha, \beta)$ refers to $R(\alpha, \beta_{[m+i]}) = R(\alpha, \beta_0, \ldots, \beta_{m+i-1})$.

We will extensively use the following simple property of the τ operator.

▶ **Proposition 4.** *Let α be a field element and $\beta = (\beta_0, \ldots, \beta_i, \ldots, \beta_{s-1}) \in \mathbb{F}^s$. Then, for any integer j with $1 \leq j \leq s - m$,*

$$\tau^{(j)}[(X - \alpha)Q](\alpha, \beta) = j\tau^{(j-1)}(Q)(\alpha, \beta).$$

Proof. We prove this by induction on j. When $j = 1$, the left-hand side is $\tau[(X - \alpha)Q](\alpha, \beta)$. From the definition, this equals $[(X - \alpha)\tau Q + Q](\alpha, \beta)$. (This can be checked term by term.) The first term $(X - \alpha)\tau Q$ is zero when evaluated at (α, β) leaving $Q(\alpha, \beta)$ as required.

For the induction step, observe similarly that

$$\begin{aligned}
\tau^{(j)}[(X - \alpha)Q] &= \tau^{(j-1)}\tau[(X - \alpha)Q] \\
&= \tau^{(j-1)}[(X - \alpha)\tau Q + Q] \\
&= \tau^{(j-1)}[(X - \alpha)\tau Q] + \tau^{(j-1)}Q
\end{aligned}$$

Evaluating at (α, β), by the induction hypothesis,

$$\begin{aligned}
\tau^{(j-1)}[(X - \alpha)\tau Q](\alpha, \beta) + \tau^{(j-1)}Q(\alpha, \beta) &= (j-1)\tau^{j-2}[\tau Q](\alpha, \beta) + \tau^{(j-1)}Q(\alpha, \beta) \\
&= (j-1)\tau^{(j-1)}Q(\alpha, \beta) + \tau^{(j-1)}Q(\alpha, \beta) \\
&= j\tau^{(j-1)}Q(\alpha, \beta). \qquad \blacktriangleleft
\end{aligned}$$

We now define lattices and state the version of Minkowski's theorem for polynomial lattices.

▶ **Definition 5** (polynomial lattice). *Let $\mathcal{B} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N\}$ be a set of m-dimensional vectors of polynomials, that is, $\mathbf{v}_i \in \mathbb{F}[X]^m$ for all i. The* lattice *$\mathcal{L}_\mathcal{B}$ generated by this set over the ring $\mathbb{F}[X]$ is the set of all linear combinations*

$$\mathcal{L}_\mathcal{B} := \{f_i\mathbf{v}_1 + f_2\mathbf{v}_2 + \cdots + f_N\mathbf{v}_N \colon f_i \in \mathbb{F}[X]\}.$$

A basis *of a lattice is a set $\mathcal{B}$ of vectors of polynomials such that $\mathcal{B}$ generates the lattice in the sense of the above definition and the vectors in $\mathcal{B}$ are linearly independent over the field $\mathbb{F}(X)$ of rational functions over $\mathbb{F}$. A basis B can be viewed as an $m \times \ell$ matrix (the ℓ basis vectors written side by side as columns).*

For a full-rank lattice (i.e., when $\ell = m$), the determinant of a basis $\mathcal{B}$ refers to the determinant of this matrix Since this quantity is independent of the choice of basis, it is also referred to as the determinant of the lattice, denoted by $\det \mathcal{L}$.

Let $\beta = (\beta(1), \beta(2), \ldots, \beta(m)) \in \mathbb{F}[X]^m$ be a vector of polynomials. We define the max-degree norm as follows $\beta := \max_{j \in [m]} \deg \beta(j)$.

▶ **Theorem 6** (polynomial version of Minkowski's theorem)**.** *Let $\mathcal{L}$ be an m-dimensional polynomial lattice. Then, there is a nonzero vector $\mathbf{v}$ in the lattice $\mathcal{L}$ satisfying*

$$\deg \mathbf{v} \leq \frac{1}{m} \deg \det \mathcal{L}$$

where $\deg \mathbf{v}$ is the max-degree norm and $\det \mathcal{L}$ is the determinant of the lattice basis.

A proof can be found in [6].

▶ **Theorem 7** ([1, Theorem 2.1],[7, Theorem 9])**.** *Let B be a set of N vectors of dimension m. Let d be the maximal degree of the polynomials which are entries of the vectors in B. Then, there is an algorithm* `ShortVector` *that finds the shortest vector in $\mathcal{L}_B$, the lattice generated by B, in time $\widetilde{O}(d(N+m)^\omega)$ where ω is the exponent in the complexity of matrix multiplication.*

Here "shortest vector" is in the max-degree norm. See [6] for details.

We will also use the following off-the-shelf subroutines for fast Hermite interpolation and polynomial Chinese remainders.

▶ **Theorem 8** ([22, Algorithm 10.22])**.** *Let $m_1(X), m_2(X), \ldots, m_r(X)$ be univariate polynomials in $\mathbb{F}[x]$ with pairwise GCD equal to 1. Let $v_1(X), v_2(X), \ldots, v_r(X) \in \mathbb{F}[X]$ such that $\deg v_i < \deg m_i$ for all $i \in [r]$. Then, we can find the unique polynomial $f(X) \in \mathbb{F}[X]$ of degree less than $d = \sum_i \deg m_i$ such that $f \equiv v_i \mod m_i$ for all $i \in [r]$, in time $O(d \mathsf{poly} \log(d))$.*

3 Fast construction of differential equation

As in Guruswami–Wang, our first step is to construct an "explanation" polynomial for the received word. In this section, we show that this can be done in near-linear time. Below is the main theorem for this section.

▶ **Theorem 9.** *Let $R = \left(\alpha, \beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)}\right)_{\alpha \in S, j \in [\ell]}$ be a received word and let $m \leq s-1$ be a parameter.*

Then, for $D \leq n\ell(s-m)/m + n\ell$, there exists a nonzero polynomial $Q(X, Y_0, Y_1, \ldots, Y_m)$ of the form $Q = \widetilde{Q}(X) + \sum_i Q_i(X) \cdot Y_i$ of X-degree at most D with the following property: if $f(X) \in \mathbb{F}_{<k}[X]$ is such that for at least $(D+k)/(s-m)$ values of $\alpha \in S$, we have

$$\left(f(\alpha), f^{(1)}(\alpha), \ldots, f^{(s-1)}(\alpha)\right) = \left(\beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)}\right)$$

for some $j \in [\ell]$, then,

$$Q\left(X, f(X), f^{(1)}(X), \ldots, f^{(m)}(X)\right) \equiv 0. \tag{2}$$

Furthermore, there is a deterministic algorithm that takes as input R and returns Q as a list of coefficients that carries out $\widetilde{O}(n\mathsf{poly}(s+m+\ell))$ field operations – specifically, $\widetilde{O}(n[\ell(s-m)^2+m^2(s-m)+(s-m)m^\omega])$.

A natural way to ensure that the polynomial Q satisfies (2), is to ensure that for each $\alpha \in S$, we have the following conditions

$$\tau^{(t)}(Q)\left(\alpha, \beta_j^0, \beta_j^1, \ldots, \beta_j^{m+t}\right) = 0, \qquad \forall t \in [s-m],\ j \in [\ell]. \tag{3}$$

This will ensure that every close enough codeword satisfies (2) (see Lemma 14 for exact details). We will refer to these conditions as the "τ-conditions". The set of polynomials Q satisfying (3) are closed under addition and multiplication by a polynomial in $\mathbb{F}[X]$. In other words, they form a $\mathbb{F}[X]$-module (or equivalently a lattice with entries from the univariate ring $\mathbb{F}[X]$). Here, it will be convenient to view the polynomial $Q = \widetilde{Q}(X) + \sum_i Q_i(X)\cdot Y_i$ as an $(m+2)$-dimensional vector of the form $(\widetilde{Q}, Q_0, Q_1, \ldots, Q_m) \in \mathbb{F}[X]^{m+2}$ where the first component $\widetilde{Q}$ is the Y-free part of Q and the subsequent components are the coefficients of $Y_0, Y_1, \ldots, Y_m$ respectively. To prove Theorem 9, we will first construct a structured basis for this lattice and then use Minkowski's theorem to show that there exists a short vector in this lattice. The following proposition gives this structured basis for a sub-lattice of the lattice of polynomials satisfying the "τ-conditions".

▶ **Proposition 10.** *Let ℓ be the list size and $R = \left(\alpha, \beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)}\right)_{\alpha\in S, j\in[\ell]}$ be the received word and let $m \le s-1$ be a parameter.*

Then, there exist $m+2$ polynomials in $\mathbb{F}[X, Y_0, \ldots, Y_m]$ satisfying the τ-conditions (3) *for all $\alpha \in S$ of the following type*

$$\begin{aligned}
\widetilde{\mathbf{B}} &= \prod_{\alpha\in S}(X-\alpha)^{s-m}, \\
\mathbf{B}_i &= Y_i \cdot \prod_{\alpha\in S}(X-\alpha)^{s-m}, && \forall i,\ \text{such that } 0 \le i \le \ell-2 \\
\mathbf{B}_i &= \widetilde{C_i}(X) + \sum_{r=0}^{i-1} Y_r \cdot C_{i,r}(X) + Y_i \cdot \prod_{\alpha\in S}(X-\alpha)^{\ell-1}, && \forall i,\ \text{such that } \ell-1 \le i \le m.
\end{aligned}$$

for some polynomials $\widetilde{C_i}, C_{i,r} \in \mathbb{F}[X]$ for $\ell-1 \le i \le m$ and $0 \le r \le i-1$. Furthermore, there is a deterministic algorithm that takes as input R and returns these $(m+2)$-polynomials carrying out $\widetilde{O}(n[\ell(s-m)^2+m^2(s-m)])$ field operations.

The analogue of Proposition 10 in the list-decoding case was easier to obtain since the conditions for all the evaluation points (i.e., $\alpha \in S$) are identical. However, this is not the case when the list size ℓ is more than 1. In order to address this, we first show that a version of Proposition 10 can be proved for each $\alpha \in S$ and then combine these different bases for the different α into a single basis that works for all the α. The following proposition is the version of Proposition 10 for a single $\alpha \in S$ (which is proved in Section 3.1).

▶ **Proposition 11.** *Let $\alpha \in S$ and ℓ be the list size and let $m \le s-1$ be a parameter.*

Given ℓ possible evaluations (i.e., $\left(\beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)}\right)$ for each $j \in [\ell]$) at the point α, there exist $m+2$ polynomials in $\mathbb{F}[X, Y_0, \ldots, Y_m]$ satisfying the τ-conditions (3) *for this specific $\alpha \in S$ of the following type*

$$\begin{aligned}
\widetilde{\mathbf{B}}_\alpha &= (X-\alpha)^{s-m}, \\
\mathbf{B}_{\alpha,i} &= Y_i \cdot (X-\alpha)^{s-m}, && \forall i, \text{ such that } 0 \le i \le \ell-2 \\
\mathbf{B}_{\alpha,i} &= \widetilde{C_i}(X) + \sum_{r=0}^{i-1} Y_r \cdot C_{i,r}(X) + Y_i \cdot (X-\alpha)^{\ell-1}, && \forall i, \text{ such that } \ell-1 \le i \le m.
\end{aligned}$$

for some polynomials $\widetilde{C_i}, C_{i,r} \in \mathbb{F}[X]$ *for* $\ell-1 \le i \le m$ *and* $0 \le r \le i-1$. *Furthermore, there is a deterministic algorithm that takes as input* $\left(\beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)}\right)_{j\in[\ell]}$ *and returns these* $(m+2)$ *polynomials, carrying out* $\widetilde{O}(\ell(s-m)^2)$ *field operations.*

We then show in Section 3.2 that the bases for two disjoint sets U, V of evaluation points (i.e., $U, V \subseteq S$) can be combined into a basis for the set $W = U \cup V$ in time approximately $\widetilde{O}(\max(|U|, |V|))$. We use this procedure recursively over the bases constructed via Proposition 11 for each $\alpha \in S$ to obtain a basis that works simultaneously for all $\alpha \in S$ to yield Proposition 10.

3.1 Construction of basis for a single evaluation point

In this section, we will fix an evaluation point $\alpha \in S$ and construct a set of $m+2$ linearly independent polynomials (over the ring $\mathbb{F}[X]$) as specified in Proposition 11. More precisely, we want to show that there exist $m+2$ polynomials of the following type that satisfy the τ-conditions for α. Since α will be fixed throughout this section, we will drop α from the subscript to declutter the notation.

$$\begin{aligned}
\widetilde{\mathbf{B}} &= (X-\alpha)^{s-m}, \\
\mathbf{B}_i &= Y_i \cdot (X-\alpha)^{s-m}, && \forall i, \text{ such that } 0 \le i \le \ell-2 \\
\mathbf{B}_i &= \widetilde{C_i}(X) + \sum_{r=0}^{i-1} Y_r \cdot C_{i,r}(X) + Y_i \cdot (X-\alpha)^{\ell-1}, && \forall i, \text{ such that } \ell-1 \le i \le m.
\end{aligned}$$

for some polynomials $\widetilde{C_i}, C_{i,r} \in \mathbb{F}[X]$ for $\ell-1 \le i \le m$ and $0 \le r \le i-1$.

To this end, given a polynomial $\mathbf{B}$ of the above form and any $\beta = (\beta^{(0)}, \beta^{(1)}, \ldots, \beta^{(s-1)})$ we will find it convenient to use the following notation for the vector in $\mathbb{F}^{s-m}$ composed of evaluations of iterated τ-operator of the polynomial $\mathbf{B}$ at the point (α, β):

$$\tau(\mathbf{B}, \beta) := \tau^{(<s-m)}(\mathbf{B})(\alpha,\beta) = \begin{bmatrix} \mathbf{B}(\alpha,\beta) \\ \tau(\mathbf{B})(\alpha,\beta) \\ \tau^{(2)}(\mathbf{B})(\alpha,\beta) \\ \vdots \\ \tau^{(s-m-1)}(\mathbf{B})(\alpha,\beta) \end{bmatrix}.$$

The linearity of the τ operator shows that the function $\tau(\cdot,\cdot)$ is linear in its first argument.

Proof of Proposition 11. In the notation defined above, we want to show that $\tau(\mathbf{B}, \beta) = \mathbf{0}_{s-m}$ for all $\beta \in \{\beta_j : j \in [\ell]\}$ and each $\mathbf{B} \in \{\widetilde{\mathbf{B}}, \mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_m\}$. Clearly, this is true when $\mathbf{B} \in \{\widetilde{\mathbf{B}}, \mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_{\ell-2}\}$. In the rest of the proof, we will construct polynomials $\widetilde{C_i}, C_{i,r} \in \mathbb{F}[X]$, such that this is also true for the polynomials $\mathbf{B} = \{\mathbf{B}_{\ell-1}, \mathbf{B}_\ell, \ldots, \mathbf{B}_m\}$. We will prove this by induction on ℓ, the list size.

Base case $\ell = 1$: This is the list-decoding setting, where we have only β in the list corresponding to α, namely $\beta_0 = (\beta_0^{(0)}, \beta_0^{(1)}, \ldots, \beta_0^{(s-1)})$. The proof in this case is similar to the corresponding step in [6], which is obtained by Hermite interpolation. For each $0 \le i \le m$, we define the polynomial $\widetilde{C_i}$ to be the unique $(s-m-1)$-degree polynomial in $\mathbb{F}[X]$ such that for all $j \in [s-m]$, we have

$$\widetilde{C_i}^{(j)}(\alpha) = -\beta_0^{(i+j)}.$$

The remaining $C_{i,r}(X)$ are set to zero for all $0 \le i \le m$ and $r \in [i]$. In other words, the ith polynomial $\mathbf{B}_i := Y_i + \widetilde{C}_i(X)$ for $\ell - 1 = 0 \le i \le m$. $\mathbf{B}_i$ clearly satisfies the τ-conditions (3) for α.

Induction step $\ell \ge 2$: By induction, let us assume that for the first $(\ell-1)$ list elements $\beta_j, j \in [\ell-1]$, we have polynomials $\mathbf{B}'_i$ satisfying the τ-conditions for α, namely $\tau(\mathbf{B}'_i, \beta_j) = \mathbf{0}$ for all i such that $\ell - 2 \le i \le m$ and $j \in [\ell-1]$. We now need to construct polynomials $\mathbf{B}_i$ for $\ell - 1 \le i \le m$ such that $\tau(\mathbf{B}_i, \beta_j) = \mathbf{0}$ for all i such that $\ell - 1 \le i \le m$ and $j \in [\ell]$. Note that the number of polynomials to be constructed in the ℓ-case is one less than the $(\ell-1)$-case while the number of β's for which we need $\tau(\mathbf{B}_i, \beta_j) = \mathbf{0}$ is one more.

Fix an i such that $\ell - 1 \le i \le m$. We will construct the polynomial $\mathbf{B}_i$ from the polynomial $\mathbf{B}'_{i-1}$. Consider the polynomial $\mathbf{B}''_i := \tau\left((X-\alpha)\cdot \mathbf{B}_{i-1}\right)$. Clearly, this polynomial is of the required form $Y_i \cdot (X-\alpha)^{\ell-1} + \sum_{r\in[i]} Y_r \cdot C_{i,r}(X) + \widetilde{C}_i(X)$. By Proposition 4, we have that $\tau(\mathbf{B}''_i, \beta_j) = \mathbf{0}$ for all $j \in [\ell-1]$. Now, if $\tau(\mathbf{B}''_i, \beta_{\ell-1})$ is also $\mathbf{0}$, we can set $\mathbf{B}_i := \mathbf{B}''_i$ and be done.

Suppose $\tau(\mathbf{B}''_i, \beta_{\ell-1}) \neq \mathbf{0}$. Then, let $r_* \in [s-m]$ be the least r such that $\tau^{(<r)}(\mathbf{B}''_i)(\alpha, \beta_{\ell-1}) = \mathbf{0}_r$ but $\tau^{(r)}(\mathbf{B}''_i)(\alpha, \beta_{\ell-1}) \neq 0$. By Proposition 4, definition of $\mathbf{B}''_i$ and the fact that $s-m$ is less than the characteristic of $\mathbb{F}$, we also have $\tau^{(<r_*)}(\mathbf{B}'_{i-1})(\alpha, \beta_{\ell-1}) = \mathbf{0}_{r_*}$ but $\tau^{(r_*)}(\mathbf{B}'_{i-1})(\alpha, \beta_{\ell-1}) \neq 0$. Let us consider the $(s-m-r_*)$ polynomials $\{(X-\alpha)^t \cdot \mathbf{B}'_{i-1} \colon t \in [s-m-r_*]\}$. The t-th polynomial in this set, namely $(X-\alpha)^t \cdot \mathbf{B}'_{i-1}$ satisfies $\tau^{(<r_*+t)}((X-\alpha)^t \cdot \mathbf{B}'_{i-1})(\alpha, \beta_{\ell-1}) = \mathbf{0}_{r_*+t}$ but $\tau^{(r_*+t)}((X-\alpha)^t \cdot \mathbf{B}'_{i-1})(\alpha, \beta_{\ell-1}) \neq 0$. Hence, the $(s-m-r_*)$ vectors $\{\tau\left((X-\alpha)^t \cdot \mathbf{B}'_{i-1}, \beta_{\ell-1}\right) : t \in [s-m-r_*]\}$ form a basis for the space of all vectors in $\mathbb{F}^{s-m}$ which are zero in the first r_* coordinates. This space contains the vector $\tau(\mathbf{B}''_i, \beta_{\ell-1})$ and hence there exist scalars $c_t \in \mathbb{F}$ for $t \in [s-m-r_*]$ such that

$$\tau(\mathbf{B}''_i, \beta_{\ell-1}) = \sum_{t\in[s-m-r_*]} c_t \cdot \tau((X-\alpha)^t \cdot \mathbf{B}'_{i-1}, \beta_{\ell-1}).$$

We can now set

$$\mathbf{B}_i := \mathbf{B}''_i - \sum_{t\in[s-m-r_*]} c_t \cdot (X-\alpha)^t \cdot \mathbf{B}'_{i-1}.$$

By choice of the scalars c_t, we have that $\tau(\mathbf{B}_i, \beta_{\ell-1}) = \mathbf{0}$. Since $\tau(\mathbf{B}''_i, \beta_j) = \mathbf{0}$ and $\tau(\mathbf{B}'_{i-1}, \beta_j) = \mathbf{0}$ for all $j \in [\ell-1]$, we also have that $\tau(\mathbf{B}_i, \beta_j) = \mathbf{0}$ for all $j \in [\ell-1]$. Combining, we have that $\tau(\mathbf{B}_i, \beta_j) = \mathbf{0}$ for all $j \in [\ell]$.

It can be seen that $\mathbf{B}_i$ is of the required form $Y_i \cdot (X-\alpha)^{\ell-1} + \sum_{r\in[i]} Y_r \cdot C_{i,r}(X) + \widetilde{C}_i(X)$ since $\mathbf{B}''_i$ is of this form and each of the polynomials $(X-\alpha)^t \cdot \mathbf{B}'_{i-1}$ are of the form $\sum_{r\in[i]} Y_r \cdot C_{i,r}(X) + \widetilde{C}_i(X)$.

We turn to the time complexity. For each list element, we may have to solve a system of equations to find the scalars $\{c_t\}$. Observe that this is a triangular system in at most $s-m$ variables, and hence can be solved with $O(s-m)^2$ operations; giving an overall total of $\widetilde{O}(\ell(s-m)^2)$ operations. This concludes the proof of the proposition. ◀

3.2 Combining the bases

In the previous section, we constructed a structured basis for each $\alpha \in S$. In this section, we will show how to combine these bases recursively to obtain a single basis that works for all $\alpha \in S$. To this end, we begin by showing a general procedure to combine the bases of two $\mathbb{F}[X]$-modules to generate the basis for a related $\mathbb{F}[X]$-module.

Let S be the set of evaluation points. For each $\alpha \in S$, let $p_\alpha(X)$ be an associated monic polynomial of degree at most $(s-m)$ such that for any two distinct $\alpha, \beta \in S$, we have $\gcd(p_\alpha, p_\beta) = 1$. We will be setting $p_\alpha(X) = (X-\alpha)^{s-m}$ for multiplicity codes.

Let $\mathcal{U}, \mathcal{V}$ be two disjoint non-empty subsets of S. Define the polynomials

$$P_{\mathcal{U}}(X) := \prod_{\alpha \in \mathcal{U}} p_\alpha(X), \qquad P_{\mathcal{V}}(X) := \prod_{\alpha \in \mathcal{V}} p_\alpha(X).$$

Observe that since $\mathcal{U} \cap \mathcal{V} = \emptyset$, we have $\gcd(P_{\mathcal{U}}, P_{\mathcal{V}}) = 1$.

Let $\mathcal{M}_{\mathcal{U}}, \mathcal{M}_{\mathcal{V}}$ be two $(m+2)$-dimensional $\mathbb{F}[X]$-modules[4] with lower-triangular bases $\mathcal{B}_{\mathcal{U}} = \{\widetilde{\mathbf{U}}, \mathbf{U}_0, \mathbf{U}_1, \ldots, \mathbf{U}_m\}$ and $\mathcal{B}_{\mathcal{V}} = \{\widetilde{\mathbf{V}}, \mathbf{V}_0, \mathbf{V}_1, \ldots, \mathbf{V}_m\}$ respectively of the following type:

$$\widetilde{\mathbf{U}} = \widetilde{U}(X), \qquad \widetilde{\mathbf{V}} = \widetilde{V}(X)$$
$$\mathbf{U}_i = \widetilde{U_i}(X) + \sum_{r=0}^{i} Y_r \cdot U_{i,r}(X), \qquad \mathbf{V}_i = \widetilde{V_i}(X) + \sum_{r=0}^{i} Y_r \cdot V_{i,r}(X), \qquad \forall 0 \le i \le m$$

Furthermore, suppose these bases $\mathcal{B}_{\mathcal{U}}, \mathcal{B}_{\mathcal{V}}$ behave "nicely" with their respective polynomials $P_{\mathcal{U}}, P_{\mathcal{V}}$ in the following sense: the diagonal elements $\widetilde{U}, U_{i,i}, i \in [m]$ are nonzero factors of the polynomial $P_{\mathcal{U}}$ and similarly $\widetilde{V}, V_{i,i}, i \in [m]$ are nonzero factors of the polynomial $P_{\mathcal{V}}$.

Let $\mathcal{W} = \mathcal{U} \cup \mathcal{V}$ and $P_{\mathcal{W}} = P_{\mathcal{U}} \cdot P_{\mathcal{V}}$ be the associated polynomial with the set $\mathcal{W}$. We can construct a lower-triangular basis $\mathcal{B}_{\mathcal{W}} = \{\widetilde{\mathbf{W}}, \mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_m\}$ which behaves "nicely" with the polynomial $P_{\mathcal{W}}$ as follows:

$$\widetilde{\mathbf{W}} = \widetilde{W}(X)$$
$$\mathbf{W}_i = \widetilde{W_i}(X) + \sum_{r=0}^{i} Y_r \cdot W_{i,r}(X), \qquad \forall 0 \le i \le m$$

where the diagonal elements are given by $\widetilde{W} = \widetilde{U} \cdot \widetilde{V}$ and $W_{i,i} = U_{i,i} \cdot V_{i,i}, 0 \le i \le m$ and the non-diagonal elements $W_{i,r}(X), r \in [i]$ are the unique polynomials (by the Chinese remainder theorem) such that

$$W_{i,r} \equiv U_{i,r} \cdot V_{i,i} \mod P_{\mathcal{U}},$$
$$W_{i,r} \equiv V_{i,r} \cdot U_{i,i} \mod P_{\mathcal{V}}.$$

This basis satisfies that $\mathbf{W}_i \equiv V_{i,i}(X) \cdot \mathbf{U}_i \bmod P_{\mathcal{U}}$ and symmetrically, $\mathbf{W}_i \equiv U_{i,i}(X) \cdot \mathbf{V}_i \bmod P_{\mathcal{V}}$.

Let $\mathcal{M}_{\mathcal{W}}$ be the $\mathbb{F}[X]$-module generated by this basis $\mathcal{B}_{\mathcal{W}}$. This basis satisfies the following property.

[4] Here, we identify the polynomial $\widetilde{Q} + \sum_{i \in [m]} Y_i \cdot Q_i(X)$ with the $(m+2)$-dimensional vector $(\widetilde{Q}, Q_0, Q_1, \ldots, Q_m)$

▷ Claim 12. If $\mathbf{H} = \widetilde{H}(X) + \sum_{i\in[m]} Y_i \cdot H_i(X)$ or equivalently $(\widetilde{H}, H_0, H_1, \ldots, H_m)$ satisfies $\mathbf{H} \bmod P_{\mathcal{U}} \in \mathcal{M}_{\mathcal{U}}$ and $\mathbf{H} \bmod P_{\mathcal{V}} \in \mathcal{M}_{\mathcal{V}}$, then $\mathbf{H} \bmod P_{\mathcal{W}} \in \mathcal{M}_{\mathcal{W}}$.

Proof. Since $\mathbf{H} \bmod P_{\mathcal{U}} \in \mathcal{M}_{\mathcal{U}}$ and $\mathbf{H} \bmod P_{\mathcal{V}} \in \mathcal{M}_{\mathcal{V}}$, we have polynomials $\widetilde{A}, \widetilde{B}, A_i, B_i, 0 \le i \le m$ such that

$$\mathbf{H} \bmod P_{\mathcal{U}} = \widetilde{A}(X) \cdot \widetilde{\mathbf{U}} + \sum_{i=0}^{m} A_i(X) \cdot \mathbf{U}_i,$$
$$\mathbf{H} \bmod P_{\mathcal{V}} = \widetilde{B}(X) \cdot \widetilde{\mathbf{V}} + \sum_{i=0}^{m} B_i(X) \cdot \mathbf{V}_i.$$

Define polynomials

$$\widetilde{C}(X) := \widetilde{U}^{-1} \cdot P_{\mathcal{U}}^{-1} \cdot \widetilde{B} \bmod P_{\mathcal{V}}, \qquad \widetilde{D}(X) := \widetilde{V}^{-1} \cdot P_{\mathcal{V}}^{-1} \cdot \widetilde{A} \bmod P_{\mathcal{U}}$$
$$C_i(X) := U_{i,i}(X)^{-1} \cdot P_{\mathcal{U}}^{-1} \cdot B_i \bmod P_{\mathcal{V}}, \quad D_i(X) := V_{i,i}^{-1} \cdot P_{\mathcal{V}}^{-1} \cdot A_i \bmod P_{\mathcal{U}}, \quad i \in [m]$$

We claim that

$$\mathbf{H} = [\widetilde{C} \cdot P_{\mathcal{U}} + \widetilde{D} \cdot P_{\mathcal{V}}]\widetilde{\mathbf{W}} + \sum_{i=0}^{m} [C_i \cdot P_{\mathcal{U}} + D_i \cdot P_{\mathcal{V}}]\mathbf{W}_i \bmod P_{\mathcal{U}} P_{\mathcal{V}}.$$

Indeed, observe the right-hand side mod $P_{\mathcal{U}}$. All the terms involving $P_{\mathcal{U}}$ vanish, leaving $\widetilde{D} \cdot P_{\mathcal{V}} \cdot \widetilde{\mathbf{W}} + \sum_{i=0}^{m} D_i \cdot P_{\mathcal{V}} \cdot \mathbf{W}_i$. Simplifying using the definition of $\mathbf{W}$, we have $\widetilde{A} \cdot \widetilde{\mathbf{U}} + \sum_{i\in[m]} A_i \cdot \mathbf{U}_i$, which is $\mathbf{H} \bmod P_{\mathcal{U}}$. The case mod $P_{\mathcal{V}}$ is symmetric and the equivalence follows from the Chinese remainder theorem. ◁

For the specific setting of multiplicity codes, the polynomial $p_\alpha(X)$ will be $(X-\alpha)^{s-m}$. For any subset $\mathcal{U} \subseteq S$ of the evaluation points, we will say that the basis $\mathcal{B}_{\mathcal{U}} = \{\widetilde{\mathbf{U}}, \mathbf{U}_0, \ldots, \mathbf{U}_m\}$ is $\mathcal{U}$-good, if the following two properties are satisfied.

1. The diagonal elements of the basis satisfy

$$\widetilde{U}(X) := \prod_{\alpha\in\mathcal{U}} (X-\alpha)^{s-m},$$
$$U_{i,i}(X) := \prod_{\alpha\in\mathcal{U}} (X-\alpha)^{s-m}, \qquad 0 \le i \le \ell - 2$$
$$U_{i,i}(X) := \prod_{\alpha\in\mathcal{U}} (X-\alpha)^{\ell-1}, \qquad \ell - 1 \le i \le m$$

   where ℓ is the list size.
2. Every basis element $\mathbf{U} \in \mathcal{B}_{\mathcal{U}}$ satisfies the τ-conditions (3) for every $\alpha \in \mathcal{U}$.

Clearly, the basis generated by Proposition 11 when $\mathcal{U} = \{\alpha\}$ is a singleton set is $\{\alpha\}$-good. The following claim shows that if $\mathcal{B}_{\mathcal{U}}, \mathcal{B}_{\mathcal{V}}$ are both good, so is $\mathcal{B}_{\mathcal{W}}$.

▷ Claim 13. Let $\mathcal{U}, \mathcal{V} \subseteq S$ be two disjoint non-empty subsets of the set of evaluation points S. If $\mathcal{B}_{\mathcal{U}}$ is $\mathcal{U}$-good and $\mathcal{B}_{\mathcal{V}}$ is $\mathcal{V}$-good, then $\mathcal{B}_{\mathcal{W}}$ is $\mathcal{W}$-good.

Furthermore, there is a deterministic algorithm that when given the bases $\mathcal{B}_{\mathcal{U}}$ and $\mathcal{B}_{\mathcal{V}}$ outputs the basis $\mathcal{B}_{\mathcal{W}}$, carrying out $\widetilde{O}(\max\{|\mathcal{U}|, |\mathcal{V}|\}m^2(s-m))$ field operations.

Proof. Item 1 follows from how the diagonal elements of the basis $\mathcal{B}_{\mathcal{W}}$ are defined. The runtime bound follows from the time required for the Chinese remainder theorem (Theorem 8): for the $O(m^2)$ off-diagonal elements $W_{i,r}$, we carry out a CRT of polynomials of degree at most $\max\{|\mathcal{U}|, |\mathcal{V}|\}(s-m)$.

We now prove Item 2. Let $\mathbf{W} \in \mathcal{B}_{\mathcal{W}}$. If $\mathbf{W} = \widetilde{\mathbf{W}}$, then $\mathbf{W} = \widetilde{\mathbf{V}} \cdot \widetilde{\mathbf{U}} = \prod_{\alpha \in \mathcal{W}} (X - \alpha)^{s-m}$ and hence satisfies the τ-conditions for all $\alpha \in \mathcal{W}$. Suppose $\mathbf{W} = \mathbf{W}_i$ for some $0 \leq i \leq m$. Let $\alpha \in \mathcal{W}$. Suppose $\alpha \in \mathcal{U}$. By definition of $\mathbf{W}_i$, we have $\mathbf{W}_i \equiv V_{i,i} \cdot \mathbf{U}_i \bmod P_{\mathcal{U}}$ and is hence of the form $V_{i,i} \cdot \mathbf{U}_i + \prod_{\alpha \in \mathcal{U}} (X - \alpha)^{s-m} \cdot \mathbf{H}$ for some polynomial $\mathbf{H} = \widetilde{H} + \sum_{i=0}^{m} Y_i \cdot H_i(X)$. Since $\mathcal{B}_{\mathcal{U}}$ is $\mathcal{U}$-good, $\mathbf{U}_i$ satisfies the τ-conditions with respect to α. Hence, so does $V_{i,i} \cdot \mathbf{U}_i$ and $\mathbf{W}_i$. The case when $\alpha \in \mathcal{V}$ is similar. ◁

We are now ready to prove Proposition 10.

Proof of Proposition 10. Let $\mathcal{B}_S := \{\widetilde{\mathbf{B}}, \mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_m\}$ be obtained by the following process: Divide S into two halves $\mathcal{U}$ and $\mathcal{V}$, and recursively compute bases $\mathcal{B}_{\mathcal{U}}$ and $\mathcal{B}_{\mathcal{V}}$ for them, and then combine them as indicated earlier in this section. For a singleton set, we use the basis given by Proposition 11.

It follows from Claim 13, that since the bases for the singleton sets are good, so are all the bases that are recursively constructed. Thus, the basis $\mathcal{B}_S$ has all the properties mentioned in Proposition 10. The only thing left to be verified is the running time.

As for the running time, let $T(n)$ be the running time where n is the size of the set S. We split it into two instances of size $n/2$, each of which can be solved in time $T(n/2)$. Claim 13 states that the bases for the two instances can be combined in time $\widetilde{O}(nm^2(s-m))$. Finally, Proposition 11 states that the base case when the set is a single point requires time $\widetilde{O}(\ell(s-m)^2)$. Putting all this together, we have $T(n) = 2T(n/2) + \widetilde{O}(nm^2(s-m))$ and $T(1) = \widetilde{O}(\ell(s-m)^2)$. Solving, we get $T(n) = \widetilde{O}(n[\ell(s-m)^2 + m^2(s-m)])$. ◀

3.3 Proof of Theorem 9

Before we prove the main theorem of the section, we have to connect the τ-conditions to the desired property, of close enough codewords satisfying the differential equation.

▶ **Lemma 14.** *Let $Q(X, Y_0, Y_1, \ldots, Y_m) \in \mathcal{M}_S$ have X-degree at most $D \leq n\ell(s-m)/m + n\ell$ and Y_i-degree at most 1 for every Y_i. Let $f(X) \in \mathbb{F}_{<k}[X]$ be such that for $(D+k)/(s-m)$ values of $\alpha \in S$, $(f(\alpha), f^{(1)}(\alpha), \ldots, f^{(s-1)}(\alpha)) = (\beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)})$ for some $j \in [\ell]$. Then, $Q(X, f(X), f^{(1)}(X), \ldots, f^{(m)}(X)) \equiv 0$.*

Proof. Let $P(X) = Q(X, f(X), f^{(1)}(X), \ldots, f^{(s-1)}(X))$. Since f has degree less than k and Q is linear in the Y-variables, $P(X)$ has degree less than $D + k$. The definition of the τ operator is set up such that $\tau^{(i)}(Q)(X, f(X), f^{(1)}(X), \ldots, f^{(s-1)}(X)) = \frac{d^i P}{dX^i}$. This combined with Proposition 10 implies that for an agreement point α,

$$\frac{d^i P}{dX^i}(\alpha) = [\tau^{(i)}(Q)](\alpha, f(\alpha), f^{(1)}(\alpha), \ldots, f^{(s-1)}(\alpha)) = \tau^{(i)}(Q)(\alpha, \beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)}) = 0$$

for any $i \in [s-m]$. That is, at these points, P vanishes with multiplicity at least $(s-m)$. Therefore, if the number of such agreements is at least $(D+k)/(s-m)$, P must be identically zero. ◀

We are now ready to prove the main theorem of the section.

Proof of Theorem 9. We will take Q to be a polynomial in the $\mathbb{F}[X]$-span of the polynomials $\widetilde{\mathbf{B}}, \mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_m$ given by Proposition 10. Since each of them satisfies the τ-conditions, as given by the proposition, so will Q. In addition, Q will have the form $Q = \widetilde{Q}(X) + \sum_i Q_i(X) \cdot Y_i$, inherited from the basis polynomials.

We now have to bound the degree of the coefficients. Following our previous notation, let $\mathcal{M}_S$ be span$\{\widetilde{\mathbf{B}}, \mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_m\}$. By Theorem 6, $\mathcal{M}_S$ contains a polynomial of degree at most $(\deg \det \mathcal{M}_S)/(m+2)$. Because of the lower triangular structure, to find $\deg \det \mathcal{M}_S$, we need only multiply the degrees of the terms on the diagonal.

The first ℓ terms on the diagonal are $\prod_{\alpha \in S}(X-\alpha)^{s-m}$, and the remaining $m+2-\ell$ are $\prod_{\alpha \in S}(X-\alpha)^{\ell-1}$. This gives $\deg \det \mathcal{M}_S = n\ell(s-m) + (m+2-\ell)n(\ell-1) \leq n\ell(s-m) + n(m+2)\ell$. Applying the Minkowski statement gives the required bound.

The close enough codewords statement is given by Lemma 14. This gives all the required properties of Q.

The running time is the time to construct the basis followed by the time to find a short vector in the lattice. From Proposition 10 and Theorem 7, we get a running time bound of $\widetilde{O}(n[\ell(s-m)^2 + m^2(s-m) + (s-m)m^\omega])$. ◀

4 List-recovery algorithm for multiplicity codes

Having constructed the differential equation, we now have to solve it in near-linear time. We use off-the-shelf subroutines for this. We recall the relevant algorithms before analyzing the overall list-recovery algorithm.

4.1 Fast algorithms for solving differential equations

We will invoke the following algorithms for solving differential equations from [6]. (Their paper also contains an analogous result for functional equations, for FRS codes.)

▶ **Theorem 15** ([6, Theorem 1.6]). *Let $\mathbb{F}$ be a finite field of characteristic greater than d or zero, and let*

$$Q(X, Y_0, \ldots, Y_m) = \widetilde{Q}(X) + \sum_{i=0}^{m} Q_i(X) \cdot Y_i$$

be a nonzero polynomial with X-degree at most D. Then, the set of polynomials $f(X) \in \mathbb{F}[X]$ of degree at most d that satisfy

$$Q\left(X, f, f^{(1)} \ldots, f^{(m)}\right) \equiv 0$$

is contained in an affine space of dimension at most m.

Furthermore, there is a deterministic algorithm `FastDESolver` *that when given Q as an input via its coefficient vector, and the parameter d, performs at most $\widetilde{O}((D+d)m^4)$ field operations and outputs a basis for this affine space.*

▶ Remark 16. The [6, Theorem 1.6] paper presents a complexity of $\widetilde{O}((D+d)\mathsf{poly}(m))$. On closer examination, their recursion contains a factor of at most m^4 as they solve for m different basis elements separately and each is solved using [6, Algorithm 1]. The time complexity for this step is analyzed in [6, Lemma 5.8] which their analysis shows has a dependence of at most $O(m^3)$.

These algorithms output a subspace containing the codewords. We use the "Prune" subroutine introduced by [15], which was simplified by [20], to obtain the codewords. The list-size bounds we present below are due to [20].

▶ **Theorem 17** ([15, Lemma 3.1], [20, Theorem 4.5]). *For every $\varepsilon > 0, \ell \in \mathbb{N}$, all integers $s > \frac{16\ell}{\varepsilon^2}$, degree parameter k, block length n and field $\mathbb{F}$ of characteristic zero or greater than k, the following is true.*

Define

$$L(\delta, \varepsilon, \ell, r) = \left(\frac{\ell}{\varepsilon}\right)^{O(\frac{1+\log(\ell)}{\varepsilon})} .$$

Then, for any $\gamma > 0$, there is a randomized algorithm `Prune` *that when given as input sets $S \subseteq \mathbb{F}$ and $E_\alpha \subseteq \mathbb{F}^s$, with $|E_\alpha| \leq \ell$ for every $\alpha \in S$ and $|S| = n$, and a basis for an affine space $\mathcal{A}$ of dimension at most $4\ell/\varepsilon$, outputs*

$$\mathcal{L} = \{f(X) \in \mathbb{F}_{<k}[X] \cap \mathcal{A} \mid |\{\alpha \in S \mid (f(\alpha), f^{(1)}(\alpha), \cdots, f^{(s-1)}(\alpha)) \in E_\alpha\}| > (R+\varepsilon)n\}$$

with probability at least $1-\gamma$ in time $\widetilde{O}(n)\mathsf{poly}(\log q, s, L(\delta, \varepsilon, \ell, r), \log(1/\gamma))$ and it is guaranteed that

$$|\mathcal{L}| \leq L(\delta, \varepsilon, \ell, r) .$$

4.2 Algorithm and proof of main theorem

The final algorithm proceeds in 3 steps: finding the differential equation, solving it, and pruning. Before restating and proving the main theorem, we state separately here the combination of the first two steps, as the time complexity in those parts is our main contribution.

▶ **Theorem 18.** *For all integers $\ell, m, s \in \mathbb{N}$ with $m < s$, degree parameter k, block length n and field $\mathbb{F}$ of characteristic zero or greater than k, the following is true.*

There is a deterministic algorithm that, when given as input sets $S \subseteq \mathbb{F}$ and $E_\alpha \subseteq \mathbb{F}^s$ with $|E_\alpha| \leq \ell$ for every $\alpha \in S$ and $|S| = n$, runs in time $O(n \log|\mathbb{F}|(\ell(s^2 + sm^3) + \frac{k}{n}m^4)\mathsf{polylog}(n, s, \log\log|\mathbb{F}|))$ and outputs the basis of an affine space $\mathcal{A} \subseteq \mathbb{F}[X]$ of dimension at most m such that $\mathcal{A}$ contains all polynomials $f(X)$ of degree less than k such that for at least $(\frac{n\ell+k}{n(s-m)} + \frac{\ell}{m})$ fraction of $\alpha \in S$,

$$(f(\alpha), \cdots, f^{(s-1)}(\alpha)) \in E_\alpha .$$

Proof. We begin by describing the algorithm.

Input: $R = \{(\alpha_i, \beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)})\}_{i\in[n], j\in[\ell]}, k \in \mathbb{Z}$

Task: Return the basis of an affine space $\mathcal{A}$ of dimension at most m containing all $f \in \mathbb{F}_{<k}[X]$ such that for $n\ell/m + (n\ell+k)/(s-m)$ values of i,

$$(f(\alpha_i), f^{(1)}(\alpha_i), \ldots, f^{(s-1)}(\alpha_i)) = (\beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)})$$

for some $j \in [\ell]$.

1. Construct a multivariate polynomial Q as per Theorem 9.
2. Solve the resulting differential equation using the algorithm in Theorem 15, to obtain an affine space A containing all the codewords.

Observe that by Theorem 9, we can construct a multivariate polynomial $Q(X, Y_0, \cdots, Y_m)$ of the form $Q = \widetilde{Q}(X) + \sum_i Q_i(X) \cdot Y_i$ of X-degree at most $D \leq \frac{n\ell(s-m)}{m} + n\ell$ with the following property: if $f(X) \in \mathbb{F}_{<k}[X]$ is such that for at least $\frac{1}{s-m}\left(\frac{n\ell(s-m)}{m} + n\ell + k\right) = \frac{n\ell}{m} + \frac{n\ell+k}{s-m}$ values of $\alpha \in S$, we have

$$(f(\alpha), \cdots, f^{(s-1)}(\alpha)) \in E_\alpha ,$$

then,

$$Q(X, f(X), f^{(1)}(X), \cdots, f^{(m)}(X)) \equiv 0$$

in time $\widetilde{O}(n\ell((s-m)^2 + (s-m)m^\omega))$ which is within our time complexity.

Now, Theorem 15 outputs the basis of the affine space of dimension at most m of the polynomials of degree $< k$ which are solutions of the above differential equation in time $\widetilde{O}((D+k)m^4)$. Substituting $D \leq \frac{n\ell(s-m)}{m} + n\ell$, the time complexity is within the claimed range. ◀

We are finally ready to prove the main theorem. We first recall the theorem statement.

▶ **Theorem 1** (main result for multiplicity codes). *For every $\varepsilon > 0$ and $\ell \in \mathbb{N}$, there exists $s_0 = \Theta(\ell/\varepsilon^2) \in \mathbb{N}$ such that for all $s > s_0$, degree parameter k, block length n, and fields $\mathbb{F}$ of characteristic zero or characteristic greater than k, satisfying $k < ns$, the following holds. There is a randomized algorithm that, given a set $S \subseteq \mathbb{F}$ of size n and sets $E_\alpha \subseteq \mathbb{F}^s$ with $|E_\alpha| \leq \ell$ for each $\alpha \in S$, takes*

$$O\Big(n \cdot \mathsf{poly}\Big(s, \ell, \log n, (\ell/\varepsilon)^{\frac{1+\log \ell}{\varepsilon}}\Big)\Big)$$

field operations and with high probability outputs all univariate polynomials $f(X) \in \mathbb{F}[X]$ of degree less than k such that for at least $(\frac{k}{sn} + \varepsilon)$ fraction of $\alpha \in S$,

$$(f(\alpha), f^{(1)}(\alpha), \ldots, f^{(s-1)}(\alpha)) \in E_\alpha.$$

Proof. We will first set the parameters. Recall that the agreement fraction for which Theorem 9 works is

$$\frac{n\ell + k}{n(s-m)} + \frac{\ell}{m}.$$

The rate R is k/ns. For the above to approximately equal $R + \varepsilon$, we set $s_0 = 16\ell/\varepsilon^2 + \frac{4\ell}{\varepsilon}$ and $m = 4\ell/\varepsilon$ with $s > s_0$. Thus,

$$\frac{n\ell + k}{n(s-m)} + \frac{\ell}{m} \leq \frac{\ell}{(s-m)} + \frac{k}{n(s-m)} + \frac{\varepsilon}{2} \leq \frac{\varepsilon^2}{16} + \frac{\varepsilon}{4} + R \cdot \left(\frac{s}{s-m}\right) \leq R + \frac{\varepsilon \cdot R}{4} + \frac{\varepsilon}{4} + \frac{\varepsilon^2}{16}$$

Since $\varepsilon \leq 1 - R$ and thus $\varepsilon^2 \leq \varepsilon(1-R)$, we get that

$$\frac{n\ell + k}{n(s-m)} + \frac{\ell}{m} < R + \varepsilon/2.$$

The algorithm is given below.

Input: $R = \{(\alpha_i, \beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)})\}_{i \in [n], j \in [\ell]}, k \in \mathbb{Z}$

Task: Return all $f \in \mathbb{F}_{<k}[X]$ such that for $n\ell/m + (n\ell + k)/(s-m)$ values of i,

$$(f(\alpha_i), f^{(1)}(\alpha_i), \ldots, f^{(s-1)}(\alpha_i)) = (\beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)})$$

for some $j \in [\ell]$.

1. Find an affine space of dimension at most m which contains all the required codewords using Theorem 18.
2. Using the algorithm in Theorem 17, obtain a list $\mathcal{L}$ of codewords and return it.

Let f be a polynomial such that for $n\ell/m + (n\ell + k)/(s-m)$ values of i,

$$(f(\alpha_i), f^{(1)}(\alpha_i), \ldots, f^{(s-1)}(\alpha_i)) = (\beta_j^{(0)}, \beta_j^{(1)}, \ldots, \beta_j^{(s-1)})$$

for some $j \in [\ell]$. By Theorem 18, f is contained in the output affine space. The algorithm in Theorem 17 finds f with high probability.

We saw in Theorem 18 that step 1 can be done in $\widetilde{O}(n\mathsf{poly}(s + m + \ell))$ time. Combining with the runtime of Theorem 17, we get a runtime of $\widetilde{O}\left(n\mathsf{poly}(s, \ell, (\ell/\varepsilon)^{\frac{1+\log \ell}{\varepsilon}})\right)$. ◀

References

1 Michael Alekhnovich. Linear Diophantine equations over polynomials and soft decoding of Reed-Solomon codes. *IEEE Trans. Inform. Theory*, 51(7):2257–2265, 2005. (Preliminary version in *43rd FOCS*, 2002). doi:10.1109/TIT.2005.850097.

2 Vikrant Ashvinkumar, Mursalin Habib, and Shashank Srivastava. Algorithmic improvements to list decoding of folded Reed-Solomon codes. In Kasper Green Larsen and Barna Saha, editors, *Proc. 37th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 880–898, 2026. doi:10.1137/1.9781611978971.35.

3 Siddharth Bhandari, Prahladh Harsha, Mrinal Kumar, and Madhu Sudan. Decoding multivariate multiplicity codes over product sets. *IEEE Trans. Inform. Theory*, 70(1):154–169, 2024. (Preliminary version in *53rd STOC*, 2021). doi:10.1109/TIT.2023.3306849.

4 Joshua Brakensiek, Yeyuan Chen, Manik Dhar, and Zihan Zhang. Combinatorial bounds for list recovery via discrete Brascamp–Lieb inequalities. In Aditya Bhaskara and Artur Czumaj, editors, *Proc. 58th ACM Symp. on Theory of Computing (STOC)*, pages 365–376. ACM, 2026. doi:10.1145/3798129.3800756.

5 Yeyuan Chen and Zihan Zhang. Explicit folded Reed-Solomon and multiplicity codes achieve relaxed generalized singleton bound. In Michal Koucký and Nikhil Bansal, editors, *Proc. 57th ACM Symp. on Theory of Computing (STOC)*, pages 1–12, 2025. doi:10.1145/3717823.3718114.

6 Rohan Goyal, Prahladh Harsha, Mrinal Kumar, and Ashutosh Shankar. Fast list-decoding of univariate multiplicity and folded Reed-Solomon codes. In Santosh Vempala, editor, *Proc. 65th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 328–343, 2024. doi:10.1109/FOCS61266.2024.00028.

7 Somit Gupta, Soumojit Sarkar, Arne Storjohann, and Johnny Valeriote. Triangular x-basis decompositions and derandomization of linear algebra algorithms over $k[x]$. *J. Symb. Comput.*, 47(4):422–453, 2012. doi:10.1016/J.JSC.2011.09.006.

8 Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Trans. Inform. Theory*, 54(1):135–150, 2008. (Preliminary version in *38th STOC*, 2006). doi:10.1109/TIT.2007.911222.

9 Venkatesan Guruswami and Madhu Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes. *IEEE Trans. Inform. Theory*, 45(6):1757–1767, 1999. (Preliminary version in *39th FOCS*, 1998). doi:10.1109/18.782097.

10 Venkatesan Guruswami and Carol Wang. Linear-algebraic list decoding for variants of Reed-Solomon codes. *IEEE Trans. Inform. Theory*, 59(6):3257–3268, 2013. (Preliminary version in *26th IEEE Conference on Computational Complexity*, 2011 and *15th RANDOM*, 2011). doi:10.1109/TIT.2013.2246813.

11 Fernando Granha Jeronimo, Tushant Mittal, Shashank Srivastava, and Madhur Tulsiani. Explicit codes approaching generalized Singleton bound using expanders. In Michal Koucký and Nikhil Bansal, editors, *Proc. 57th ACM Symp. on Theory of Computing (STOC)*, pages 843–854, 2025. doi:10.1145/3717823.3718302.

12 Fernando Granha Jeronimo and Nikhil Shagrithaya. Probabilistic guarantees to explicit constructions: Local properties of linear codes. In Aditya Bhaskara and Artur Czumaj, editors, *Proc. 58th ACM Symp. on Theory of Computing (STOC)*, pages 806–813, 2026. doi:10.1145/3798129.3800795.

13 Itay Kalev and Amnon Ta-Shma. Unbalanced expanders from multiplicity codes. In Amit Chakrabarti and Chaitanya Swamy, editors, *Proc. 26th International Conf. on Randomization and Computation (RANDOM)*, volume 245 of *LIPIcs*, pages 12:1–12:14. Schloss Dagstuhl, 2022. doi:10.4230/LIPIcs.APPROX/RANDOM.2022.12.

14 Swastik Kopparty. Some remarks on multiplicity codes. In Alexander Barg and Oleg R. Musin, editors, *Discrete Geometry and Algebraic Combinatorics*, volume 625 of *Contemporary Mathematics*, pages 155–176. AMS, 2014. doi:10.1090/conm/625/12497.

15 Swastik Kopparty, Noga Ron-Zewi, Shubhangi Saraf, and Mary Wootters. Improved list decoding of Folded Reed-Solomon and Multiplicity codes. *SIAM J. Comput.*, 52(3):794–840, 2023. (Preliminary version in *59th FOCS*, 2018). doi:10.1137/20M1370215.

16 Nicolas Resch and S. Venkitesh. List recoverable codes: The good, the bad, and the unknown (hopefully not ugly). (manuscript).

17 Shashank Srivastava. Improved list size for folded Reed-Solomon codes. In Yossi Azar and Debmalya Panigrahi, editors, *Proc. 36th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 2040–2050, 2025. doi:10.1137/1.9781611978322.64.

18 Shashank Srivastava and Madhur Tulsiani. List decoding expander-based codes up to capacity in near-linear time. In Rotem Oshman and Ran Raz, editors, *Proc. 66th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 1465–1487, 2025. doi:10.1109/FOCS63196.2025.00077.

19 Madhu Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *J. Complexity*, 13(1):180–193, 1997. (Preliminary version in *37th FOCS*, 1996). doi:10.1006/jcom.1997.0439.

20 Itzhak Tamo. Tighter list-size bounds for list-decoding and recovery of folded Reed-Solomon and multiplicity codes. *IEEE Trans. Inform. Theory*, 70(12):8659–8668, 2024. doi:10.1109/TIT.2024.3402171.

21 Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.

22 Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 3 edition, 2013. doi:10.1017/CBO9781139856065.

23 Mary Wootters. CS250/EE387: Algebraic error correcting codes, 2019. (A course on coding theory at Stanford, Winter 2019). URL: https://web.stanford.edu/~marykw/classes/CS250_W19/index.html.

Algorithmic Phase Transition for Large Independent Sets in Dense Hypergraphs

Abhishek Dhawan
Department of Mathematics, University of Illinois Urbana-Champaign, IL, USA

Nhi U. Dinh
Department of Mathematics, University of Illinois Urbana-Champaign, IL, USA

Eren C. Kızıldağ
Department of Statistics, University of Illinois Urbana-Champaign, IL, USA

Neeladri Maitra
Department of Mathematics, University of Illinois Urbana-Champaign, IL, USA

Bayram A. Şahin
Department of Statistics, University of Illinois Urbana-Champaign, IL, USA

Abstract

We study the algorithmic tractability of finding large independent sets in dense random hypergraphs. In the sparse regime, much of the natural algorithms can be formulated within either the local or the low-degree polynomial (LDP) framework, and a rich literature has subsequently identified nearly sharp algorithmic thresholds within these classes by exploiting their stability. In the dense setting, however, the algorithmic paradigms are fundamentally different: they are online and thus need not be stable. Perhaps more crucially, even for the classical Erdős–Rényi random graph $G(n,p)$, LDPs are conjectured to fail in the "easy" regime accessible to online algorithms, thereby challenging their viability for dense models.

Our focus is on two models: (i) finding large independent sets in dense r-uniform Erdős–Rényi hypergraphs, where each size-r hyperedge is present independently with probability p, and (ii) the more challenging problem of finding large γ-balanced independent sets in dense r-uniform r-partite hypergraphs, where the vertex set is the disjoint union $V_1 \sqcup \cdots \sqcup V_r$ with $|V_i| = n$ for all i, each hyperedge in $V_1 \times \cdots \times V_r$ is present independently with probability p, and the i-th coordinate of $\gamma \in \mathbb{Q}^r$ specifies the proportion of vertices from V_i in the independent set. For both models, we pinpoint the size of the largest independent set and design online algorithms that achieve a multiplicative approximation factor of $r^{1/(r-1)}$ in the uniform and $(\max_i \gamma_i)^{-1/(r-1)}$ in the r-partite model. Furthermore, we establish matching algorithmic lower bounds, showing that these computational gaps are *sharp*: no online algorithms can breach these gaps.

Our results provide a detailed landscape for dense hypergraphs, thereby completing the picture for dense models in a manner parallel to the sparse counterparts developed recently. Our main technical contribution is twofold: a novel *staged and bucketed greedy* algorithm and a stopping-time argument tailored to the hypergraph and multipartite structure for algorithmic hardness, both of which may be of independent interest. The algorithms and proof techniques in the dense regime differ substantially from those in the sparse, yet the resulting computational gaps are remarkably analogous, pointing to a form of universality. More conceptually, our results corroborate a hypothesis from statistical mechanics linking glassy equilibrium to computational hardness: optimization problems become far more intricate in the presence of global constraints.

2012 ACM Subject Classification Theory of computation → Randomness, geometry and discrete structures; Theory of computation → Online algorithms; Mathematics of computing → Combinatorial optimization; Mathematics of computing → Hypergraphs; Mathematics of computing → Random graphs

Keywords and phrases independent sets, random hypergraphs, online algorithms, overlap gap property, statistical–computational gap, Erdős–Rényi hypergraph

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.68



© Abhishek Dhawan, Nhi U. Dinh, Eren C. Kızıldağ, Neeladri Maitra, and Bayram A. Şahin;
licensed under Creative Commons License CC-BY 4.0

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2026).
Editors: Mohit Singh and Tom Gur; Article No. 68; pp. 68:1–68:13



Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2605.05618 [17]

Funding *Abhishek Dhawan*: Partially supported by the NSF RTG grant DMS-1937241.

1 Introduction

In this paper, we study the statistical and algorithmic landscape of large independent sets in two hypergraph models: (i) the r-uniform Erdős–Rényi hypergraph $H_r(n,p)$, where each size-r subset of $[n] \coloneqq \{1,\dots,n\}$ is present independently with probability p, and (ii) the r-uniform r-partite hypergraph $H(r,n,p)$, whose vertex set is partitioned into r parts $V_1,\dots,V_r$, each of size n, where each edge in $V_1 \times \cdots \times V_r$ is present independently with probability p. Our focus is on the *dense* regime in which p is a constant (as $n \to \infty$). While the landscape for sparse random hypergraphs has been characterized in detail recently [21], the dense regime has remained elusive and requires different techniques. We examine large independent sets in $H_r(n,p)$ and large *balanced* independent sets in $H(r,n,p)$, where the selected vertices are required to be distributed across the r parts according to a prescribed vector $\gamma \in \mathbb{Q}_+^r$. For both models, we pinpoint the largest possible size – referred to as the *statistical threshold*, design polynomial-time algorithms that provably outputs a large independent set, and establish sharp algorithmic lower bounds. Taken together, our results reveal *statistical-computational gaps* analogous to those in the sparse regime, thereby pointing to a form of universality.

Establishing sharp results for dense hypergraphs requires new technical ingredients. To this end, we introduce a novel *staged and bucketed greedy* algorithm tailored to hypergraphs and design a geometrical barrier, equipped with a carefully chosen stopping time, to establish tight algorithmic lower bounds. Informally, the *staged and bucketed* greedy procedure is divided into r stages where in each stage, the algorithm populates the independent set from an "unexhausted" part until a certain target capacity is reached, and then locks that part. As for our algorithmic barrier, it is adapted to the multipartite structure and global balancedness constraint; specifically, it tracks the first time at which the independent set contains sufficiently many vertices from every part of the partition. See Section 1.3 for a more detailed overview. We expect these new tools to be of independent interest for understanding algorithmic tractability in hypergraph models and beyond.

Our algorithmic viewpoint is *online*: vertices are exposed sequentially, and upon the arrival of a vertex v, the algorithm learns the status of hyperedges involving v together with previously exposed vertices, and must irrevocably decide whether to include v in its output. As we explain below, online algorithms are central in the dense setting, where they capture the only known class of polynomial-time algorithms. This stands in stark contrast to the sparse setting, where the predominant algorithmic frameworks are markedly different, see Section 1.1.

We emphasize that, while approximating the largest independent set to within a factor of $n^{1-\epsilon}$ is already NP-hard in the worst-case [38, 47], the average-case picture is substantially richer. For the Erdős–Rényi random graph $G(n,\frac{1}{2})$, the independence number – size of the largest independent set – is approximately $2\log_2 n$, whereas the online greedy algorithm reaches $\log_2 n$, both with high probability (whp) as $n \to \infty$ [11, 37, 54, 55]. In his seminal 1976 work [46], Karp challenged the community to design an efficient algorithm that reaches $(1+\epsilon)\log_2 n$ (whp) for some $\epsilon > 0$, or to prove that this is impossible (modulo a complexity-theoretic assumption such as $P \neq NP$). Nearly five decades later, this question remains open

and is widely regarded as a cornerstone of average-case complexity and random graph theory. It has inspired several major developments – such as Jerrum's introduction of the *planted clique* problem [44], and a substantial literature has since uncovered analogous "factor-2 gaps" across a broad range of random graphs models.[1] For an overview of such gaps in random graphs and beyond, see the surveys [29, 30, 34, 7, 63, 27].

1.1 Comparison between Online Algorithms and Low-Degree Polynomials

There is by now a substantial literature on algorithmic phase transitions for the largest independent set problem in sparse random graphs (including the classical Erdős–Rényi model as well as bipartite and multipartite analogues), where nearly sharp lower bounds have been established for local algorithms and low-degree polynomials (LDPs) [35, 60, 62, 59, 21, 31]. In the sparse regime, this framework is particularly natural: local neighborhoods remain small and approximately tree-like, and several natural algorithms can be formulated within either the local or low-degree framework [62, 59, 21, 31].

▶ Remark 1. We note that in the low-degree setting, the local tree-like property is essential to control the variance of certain quantities. We discuss this further in relation to our work below.

The dense regime, however, is fundamentally different. Even for the classical Erdős–Rényi graph $G(n,p)$ with constant p, it is likely that the online greedy algorithm cannot be implemented as an LDP. An informal justification for this is through a certain *invariant*: while LDPs enjoy input stability [62, 31, 59], the online greedy algorithm is shown to be unstable [32], thus unlikely to be implemented as an LDP. The current state of affairs is in fact more subtle. A remarkable conjecture put forth in a recent AIM workshop [1] suggests that LDPs fail even in the "easy" regime:

▶ **Conjecture 2.** *In $G(n,\frac{1}{2})$, no degree-$o(\log^2 n)$ polynomial can find an independent set of size $0.9\log_2 n$.*

Accordingly, while LDPs provide an important proxy for efficient algorithms in sparse random graphs, their viability in the dense setting, even for the classical Erdős–Rényi random graph $G(n,p)$ with constant p, is questionable.[2] If true, Conjecture 2 would show that LDPs are actually not viable for dense random graphs. A relatively weaker variant of this conjecture was proven in the aforementioned workshop [1]. This motivates the study of online algorithms in the dense regime.

▶ Remark 3. The online greedy algorithm is in fact relevant well beyond the dense setting. Indeed, in the intermediate regime where $p = o(1)$ and $np \to \infty$, the independence number of $G(n,p)$ is $(2+o(1))\log(np)/p$, whereas the greedy algorithm finds an independent set of size $(1+o(1))\log(np)/p$, both whp [10, 28, 25]. The same phenomenon persists in the sparse regime: in the double limit $n \to \infty$ followed by $d \to \infty$, the independence number of $G(n,d/n)$ is $(2+o_d(1))n\log d/d$, while greedy reaches $(1+o_d(1))n\log d/d$ [13]. Thus, throughout dense, intermediate, and sparse regimes, the online greedy yields 1/2-approximation to the optimum (at the level of the leading-order terms).

[1] Importantly, Karp's original conjecture regarding "factor-2 gaps" was formulated for the dense regime, while much of the recent effort has been focused on the sparse setting.

[2] Degree-$o(\log^2 n)$ polynomials are used as a proxy for polynomial-time algorithms; see [40, 51, 63].

Lower bounds for computational intractability of finding large independent sets in $G(n,p)$ within the framework of online algorithms had remained elusive, emerging only very recently [32].[3] At a technical level, they required substantial refinements of the machinery based on the Overlap Gap Property (see below for details). These results were recently extended to the balanced independent set problem in dense bipartite random graphs [19]. As we detail below, the largest balanced independent set problem in bipartite models closely resemble their classical Erdős–Rényi counterparts (even though the unconstrained independent set problem in bipartite random graphs is polynomial-time solvable via a max-flow reduction). As such, a variant of Conjecture 2 plausibly holds for bipartite models – this was in fact put forth in [19, Conjecture 1.6].

Bipartite Random Graphs. For the large balanced independent set problem in sparse random bipartite graphs, Perkins and Wang [59] obtained sharp algorithmic results within both the local and LDP frameworks. Their algorithm arises from bipartiteness, but the sparsity is crucial in placing it into the local or LDP framework – their analysis breaks down in the absence of sparsity. They explicitly note that, since sparse random graphs are locally tree-like, the performance of local algorithms is determined, to first order, by evaluation on a Galton-Watson tree. The same is true also in the bipartite case. More specifically, the analysis of the local algorithm uses the fact that for sparse random graphs, the neighborhoods are asymptotically tree-like and hence can be approximated by a Galton-Watson tree, which they leverage to control certain acceptance probabilities and to bound neighborhood sizes; see [59, Section 2.1]. As for the low-degree analysis, one must control the variance of a certain LDP, for which it is essential to control the size of certain neighborhoods. In [59, Section 2.2], this is handled precisely through a Poisson approximation, which unfortunately holds only in the sparse regime and is no longer valid for the dense setting.

For these reasons, results of [59] for sparse bipartite graphs do not extend to the dense setting. In a dense bipartite graph, even a depth-1 neighborhood already contains $\Theta(n)$ vertices (along with substantial dependencies), so the local limit argument based on Galton-Watson approximation disappears. Consequently, if one tries to adapt the algorithmic framework of [59], the resulting polynomial will unfortunately have a large variance, breaking down the analysis. As for the balanced independent set problem in sparse random multipartite hypergraphs, recent work of Dhawan and Wang [21] devised LDP algorithms, for which the analysis does not transfer to dense regime either.

For these models, the main source of hardness is not multipartiteness per se, but rather the *balancedness* constraint. More broadly, once one imposes a global constraint such as balancedness, the resulting optimization problem becomes substantially more delicate.[4] In particular, the largest balanced independent set problem in dense bipartite or multipartite random graphs is much closer in flavor to the classical largest independent set problem in dense $G(n,p)$ than to the unconstrained independent set problem on bipartite graphs. In light of this, online algorithms offer a natural algorithmic framework in the dense regime, across both the Erdős–Rényi and multipartite settings, whereas the viability of the low-degree framework (or local algorithms) is presently much less clear.[5]

[3] We note, however, that lower bounds for LDPs appeared earlier in [43, 62].

[4] In the language of statistical mechanics, a global constraint such as balancedness is expected to induce "glassy equilibrium" and lead to computational hardness [58, 56, 59, 34]; this has been rigorously verified for certain models, see, e.g., [22, 23].

[5] For related reasons, online algorithms also play an important role in other optimization problems such as coloring over dense random graphs; see [49].

Extension to Hypergraphs and Multipartite Models. The higher uniformity ($r \geq 3$), the multipartite structure, and the balancedness constraint collectively make the rigorous study of random hypergraph models substantially more delicate. To address these challenges, we develop new technical ideas both on the algorithmic side and for establishing sharp algorithmic hardness. These ideas contribute to the technical toolkit for dense random structures, which has only recently begun to emerge; we expect them to be broadly useful for hypergraph models and beyond.

▶ Remark 4. While it is well-established that the independence number of various pseudorandom graph classes attains the computational threshold – a line of inquiry originating with Shearer's foundational work on K_3-free graphs from 1983 [61] and expanding into a robust body of literature [18, 6, 48, 15] – an analogous result for pseudorandom hypergraphs was entirely absent from the literature until the recent work of Dhawan, Methuku, and Vo [20]. There are, however, several earlier results that match the growth rate [4, 26, 16].

These results suggest that finding large independent sets in hypergraphs is inherently harder than the graph case, further motivating the study of hypergraphs.

For a detailed technical overview, see Section 1.3. A summary of our contributions is in order.

1.2 Summary of Main Results

We now provide informal versions of our main results. Recall the descriptions of our models: the first is the dense r-uniform Erdős–Rényi hypergraph $H_r(n,p)$, where each r-element subset of $[n] \coloneqq \{1,\dots,n\}$ is present independently with probability p; the second is the dense r-uniform r-partite hypergraph $H(r,n,p)$, where the vertex set is partitioned into r parts $V_1,\dots,V_r$ of size n and each r-tuple from $V_1 \times \cdots \times V_r$ is included independently with probability p.

For $b \coloneqq \frac{1}{1-p}$ and $\gamma \coloneqq (\gamma_1,\dots,\gamma_r) \in \mathbb{Q}_+^r$ with $\sum_i \gamma_i = 1$, define

$$\alpha_{\mathtt{r}}^{\mathrm{STAT}} \coloneqq (r! \log_b n)^{\frac{1}{r-1}} \qquad \text{and} \qquad \alpha_{\mathtt{r\text{-}bal}}^{\mathrm{STAT}} \coloneqq \left(\frac{\log_b n}{\prod_{i=1}^r \gamma_i}\right)^{\frac{1}{r-1}}.$$

Throughout this paper, we focus on constant p, $p = \Theta(1)$ (as $n \to \infty$). We ignore all floor/ceiling operators for simplicity with the understanding that this has no effect on our arguments.

Our first result pins down the size of the largest (γ-balanced) independent set.

▶ **Theorem 5.** *For $H_r(n,p)$, the largest independent set has size $(1 \pm o(1))\,\alpha_{\mathtt{r}}^{\mathrm{STAT}}$ whp. Likewise, for $H(r,n,p)$, the largest γ-balanced independent set has size $(1 \pm o(1))\,\alpha_{\mathtt{r\text{-}bal}}^{\mathrm{STAT}}$ whp.*

Theorem 5 identifies $\alpha_{\mathtt{r}}^{\mathrm{STAT}}$ and $\alpha_{\mathtt{r\text{-}bal}}^{\mathrm{STAT}}$ as the *statistical threshold* for independent sets in $H_r(n,p)$ and γ-balanced independent sets in $H(r,n,p)$, respectively. The proof of Theorem 5 is based on the second moment method and is, therefore, non-constructive. This naturally leads to an algorithmic question: can we efficiently find large independent sets in $H_r(n,p)$ or large γ-balanced independent sets in $H(r,n,p)$? To address this question, we let

$$\alpha_{\mathtt{r}}^{\mathrm{COMP}} \coloneqq ((r-1)! \log_b n)^{\frac{1}{r-1}} \qquad \text{and} \qquad \alpha_{\mathtt{r\text{-}bal}}^{\mathrm{COMP}} \coloneqq \left(\frac{\log_b n}{\prod_{i=1}^r \gamma_i} \max_{1 \leq i \leq r} \gamma_i\right)^{\frac{1}{r-1}}.$$

Our next result establishes that the thresholds $\alpha_{\mathtt{r\text{-}bal}}^{\mathrm{COMP}}$ and $\alpha_{\mathtt{r}}^{\mathrm{COMP}}$ are achievable in polynomial time via an online algorithm.

▶ **Theorem 6.** *For $H_r(n,p)$, there is an online algorithm which finds, for any $\epsilon > 0$, an independent set of size $(1-\epsilon)\alpha_r^{\mathrm{COMP}}$ whp. Likewise, for $H(r,n,p)$, there is an online algorithm which finds, for any $\epsilon > 0$, a γ-balanced independent set of size $(1-\epsilon)\alpha_{r\text{-}bal}^{\mathrm{COMP}}$ whp.*

We now compare the statistical thresholds with the algorithmically achievable values. For the dense r-uniform hypergraph $H_r(n,p)$, there is a factor-$r^{1/(r-1)}$ gap between the two, which recovers the well-known "factor-2 gap" for the special case $r = 2$. As for $H(r,n,p)$, the gap is of the order $(\max_{1\le i\le r}\gamma_i)^{-1/(r-1)}$. (Note that for $\gamma_1 = \cdots = \gamma_r = 1/r$, the gaps in both cases are identical.)

Are there fundamental computational barriers in these models? Our final result addresses this question and yields sharp algorithmic lower bounds, suggesting that the barriers above are not spurious and likely inherent.

▶ **Theorem 7.** *For $H_r(n,p)$ and any $\epsilon > 0$, no online algorithm finds an independent set of size $(1+\epsilon)\alpha_r^{\mathrm{COMP}}$ with probability $o(1)$. Likewise, for $H(r,n,p)$ and any $\epsilon > 0$, no online algorithm finds a γ-balanced independent set of size $(1+\epsilon)\alpha_{r\text{-}bal}^{\mathrm{COMP}}$ with probability $o(1)$.*

Note that aside from onlineness, there is no restriction on the algorithms ruled out by Theorem 7. In particular, so long as the algorithm is online, we impose no restrictions on its runtime. Importantly, Theorem 7 rules out algorithms succeeding even with a vanishing probability $o(1)$. In relevant literature, this is known as *strong hardness*; see [43] as well as [32, 19].

▶ Remark 8. More explicitly, the $o(1)$ guarantee is of the form $\exp(-\Omega(\alpha_r^{\mathrm{COMP}}\log n))$ for $H_r(n,p)$ and of form $\exp(-\Omega(\alpha_{r\text{-}bal}^{\mathrm{COMP}}\log n))$ for $H(r,n,p)$. Both of these guarantees are essentially optimal, since the probability that a randomly chosen subset with size reaching the computational threshold is independent in $H_r(n,p)$ (or a γ-balanced independent set in $H(r,n,p)$) is of the same order.

Taken together, Theorems 6 and 7 sharply characterize the performance of online algorithms in dense r-uniform hypergraphs and dense r-partite hypergraphs, suggesting an inherent computational barrier, precisely at α_r^{COMP} and $\alpha_{r\text{-}bal}^{\mathrm{COMP}}$. Our result recovers the analogous computational gaps present for $r = 2$, generalizing them to $r > 2$, thus pointing towards a certain universality.[6]

▶ Remark 9. Note that for $H_r(n,p)$, the computational gap is of the order $r^{1/(r-1)}$, which vanishes as $r \to \infty$. Likewise, the same conclusion also holds for $H(r,n,p)$, using the fact $\max_i \gamma_i \ge 1/r$. Thus, the computational problem gets "easier" when the sizes of the hyperedges become large.

▶ Remark 10. As discussed below, we introduce a novel variant of the greedy algorithm – staged and bucketed greedy – together with a carefully designed stopping time that accounts for both the multipartite structure and the balancedness constraint. Both ingredients are crucial for addressing hypergraphs and multipartite structures, and for moving beyond the classical $G(n,p)$ and bipartite setting. In this sense, our results substantially generalize several existing results in the literature. Indeed, for $H_r(n,p)$ with $r = 2$, we recover both Karp's algorithmic result [46] as well as the lower bound of [32]. Likewise, for $H(r,n,p)$, our results with $r = 2$ recover the full set of results in [19].

[6] We remark that such a universality arises also in the context of sparse random graphs, see [59, 21, 62].

1.3 Proof Overview

Our proofs for $H_r(n,p)$ and $H(r,n,p)$ follow a consistent three-part strategy: identifying the statistical threshold, demonstrating achievability via online algorithms, and applying OGP to establish matching algorithmic lower bounds. We focus first on $H_r(n,p)$ to clarify the underlying mechanics. We then discuss the technical innovations required for the more complex $H(r,n,p)$ setting, including the *staged and bucketed* greedy procedure and a new stopping time argument, both of which account for the multipartite structure and the global balancedness requirement.

The statistical threshold is established via the second moment method [5]. As for the algorithms, our approach is through an online greedy algorithm. To the best of our knowledge, ours is the first work to analyze the greedy independent set procedure on Erdős–Rényi hypergraphs for $r \geq 3$.

Our impossibility result is positioned in a body of work initiated by Gamarnik and Sudan [35], who investigated local algorithms for independent sets in sparse random graphs using the Overlap Gap Property (OGP); see below for a background. This property yields a dichotomy: the intersection of two "large" independent sets must be either significantly large or very small, creating a forbidden "gap" in overlap sizes. They demonstrated that a successful local algorithm could be leveraged to produce independent sets that violate this OGP. While this approach was refined by Rahman and Virág [60] and extended by Wein [62] to LDPs, these methods fail in dense settings. For dense random graphs, LDPs – and local algorithms, which can be implemented as LDPs – are conjectured to fail; see Conjecture 2. By contrast, the only known algorithmic frameworks that achieve large independent sets in this regime are online. The arguments of [62, 60] crucially rely on algorithmic stability, whereas online algorithms are known to be unstable [32]. Thus, the dense setting calls for substantially different techniques.

In a recent development, Gamarnik, Kızıldağ, and Warnke [32] introduced an OGP-based framework tailored for the online setting. This approach is inherently algorithm-dependent: it utilizes a stopping time to monitor output size, together with interpolation paths that evolve temporally as the algorithm progresses. It yielded sharp lower bounds within online algorithms for the classical Erdős–Rényi random graph $G(n,p)$. Extending these techniques to higher uniformities and multipartite models under a global balancedness constraint turns out to be substantially more challenging, due to additional technical hurdles that are absent in $G(n,p)$ (discussed below). Among other technical contributions, we introduce a new stopping-time argument tailored specifically to hypergraphs, which may be of independent interest. Taken together, our techniques allow us to complete the picture for hypergraph models: while the sparse regime was recently addressed by Dhawan and Wang [21], the dense regime had remained open, which is our focus.

Formally, we construct a family of correlated random hypergraphs that remain identical until a specific random stopping time, after which they evolve independently. Executing the algorithm on these correlated instances yields a collection of candidate independent sets with rigid structural constraints – most notably, they must coincide on all vertices revealed prior to the stopping time. We define these configurations as "forbidden tuples" and prove they are absent from the correlated random hypergraphs with high probability. Since a successful online algorithm would necessarily imply the existence of such tuples, this contradiction establishes our lower bound.

We next extend our analysis to the $H(r,n,p)$ model. Although the statistical threshold is determined via a second-moment argument, the γ-balancedness, a global constraint, adds substantial technical complexity. As for developing an online algorithm, the primary hurdle

in this model is the global nature of the balancedness requirement, which must be maintained throughout the construction. To address this, we introduce a *staged and bucketed* greedy algorithm. The procedure is partitioned into r distinct stages; in each stage, the algorithm populates the independent set using vertices from "unexhausted" parts until some such part reaches its target capacity and is "locked." By controlling acceptance probabilities and bounding failure events via careful estimates, we prove that this algorithm yields a γ-balanced independent set of size $(1-\epsilon)\alpha^{\text{COMP}}_{\texttt{r-bal}}$ with high probability.

The derivation of the impossibility result for $H(r,n,p)$ is substantially more involved. The main difficulties stem from two sources: (i) the multipartite nature of $H(r,n,p)$, together with the permutation invariance of the parts in the partition, and (ii) the γ-balancedness constraint, which imposes prescribed cardinality requirements on the intersections of the independent set with the r parts. Among other technical ideas, we introduce a novel stopping time that accounts for both issues simultaneously. Specifically, we stop when the independent set contains sufficiently many vertices from all parts of the partition (by contrast, the stopping time for $H_r(n,p)$ is triggered once the algorithm's output reaches a prescribed size threshold).

Background on OGP. As mentioned earlier, our algorithmic lower bound is based on the OGP framework pioneered by Gamarnik and Sudan [35, 36] (and termed in [33]). The OGP is among the most powerful techniques for identifying computationally hard phases of random optimization problems – i.e., those in which the objective function is random. Its origins lie in earlier works on random CSPs, where the apparent hard phases were observed to exhibit striking geometric phase transitions [3, 2, 57]. The OGP framework systematically converts such geometric features of the solution landscape into formal algorithmic lower bounds; see [29, 32] for surveys on the method.

We now describe the OGP in greater detail for random graphs – the context in which it was originally introduced. For the sparse Erdős–Rényi model $G(n,\frac{d}{n})$, the work of Gamarnik and Sudan [35, 36] established that independent sets of size greater than $(1+1/\sqrt{2})n\frac{\log d}{d}$ must either substantially overlap or be nearly disjoint.[7] As a consequence, they rigorously ruled out local algorithms at this threshold, thereby refuting a well-known conjecture by Hatami, Lovász, and Szegedy [39], widely believed at the time. A subsequent work by Rahman and Virág [60] extended this threshold down to $n\frac{\log d}{d}$ by analyzing the overlap patterns of many independent sets, below which polynomial-time algorithms are known [52]. This approach, now known as the multi OGP (m-OGP), has since proved instrumental in obtaining sharp algorithmic lower bounds in a variety of other models. In the context of random graphs, hardness at the $(1+1/\sqrt{2})n\frac{\log d}{d}$ threshold was later extended to LDPs [31], and sharp lower bounds for LDPs were subsequently obtained in [62] using an asymmetric variant of the m-OGP. Still more delicate refinements of the m-OGP were required in other models, such as random k-SAT [12] and spin glasses [41, 42, 43]. For the latter, this led to the branching OGP, an especially powerful variant organized around an ultrametric tree of solutions. Even more recently, novel variants of the OGP tailored for online algorithms – which need not be stable – have begun to emerge [32, 19].

A summary of our contributions along with potential future directions of inquiry are now in order.

[7] For sparse random graphs, the relevant asymptotic regime is the double limit $n\to\infty$ followed by $d\to\infty$.

1.4 Summary and Open Problems

In this work, we consider the computational hardness of finding large (γ-balanced) independent sets in random hypergraphs. We focus on online algorithms, one of the most natural algorithmic frameworks containing Karp's original algorithm for $G(n,p)$, which motivated his celebrated conjecture regarding statistical-computational gaps for independent sets in $G(n,p)$. Our results imply a computational gap of a multiplicative factor of $r^{1/(r-1)}$ for independent sets in $H_r(n,p)$ and a gap of a multiplicative factor of $(\max_i \gamma_i)^{-1/(r-1)}$ for γ-balanced independent sets in $H(r,n,p)$ with respect to online algorithms; these gaps match those shown for LDPs in the sparse regime [21].

We conclude this introduction with a description of potential future directions of inquiry.

Future Queries. The setting of [32] permits querying a limited set of *future edges* – edges incident to vertices not yet seen – at each step. That is, the decision at time t is based not only on the edges revealed so far, but also on a restricted set of such future edges. In this augmented model, [32] prove both lower bounds and algorithmic guarantees for independent sets in graphs: specifically, they show that algorithms with modest access to future information can in fact exceed the $(1+\epsilon)\log_b n$ threshold, albeit using quasi-polynomial time. Similarly, [19] prove analogous results for γ-balanced independent sets in Erdős–Rényi bipartite graphs.

This naturally raises the following question: can algorithms with limited future information outperform the $(1+\epsilon)\alpha_{\mathtt{r}}^{\mathrm{COMP}}$ threshold for $H_r(n,p)$ and the $(1+\epsilon)\alpha_{\mathtt{r\text{-}bal}}^{\mathrm{COMP}}$ threshold for $H(r,n,p)$? We conjecture that the answer is yes:

▶ **Conjecture 11.** *Let E be the set of all future edges ever revealed to the algorithm and denote by $\binom{\mathcal{A}(G)}{r}$ the set of all r-edges whose vertices are contained in the output $\mathcal{A}(G)$.*

1. *For any $\epsilon > 0$ and $r \in \mathbb{N}$, there exists a constant $c_{\epsilon,r} > 0$ and an online algorithm $\mathcal{A}$ that finds an independent set of size at least $(1+\epsilon)\alpha_r^{\mathrm{COMP}}$ whp in $H_r(n,p)$, provided*

$$\left|E \cap \binom{\mathcal{A}(G)}{r}\right| \le c_{\epsilon,r}\Big(\log_b(np)\Big)^{\frac{r}{r-1}}.$$

2. *For any $\epsilon > 0$ and $r \in \mathbb{N}$, there exists a constant $c_{\epsilon,r} > 0$ and an online algorithm $\mathcal{A}$ that finds a γ-balanced independent set of size at least $(1+\epsilon)\alpha_{r\text{-}bal}^{\mathrm{COMP}}$ whp in $H(r,n,p)$, provided*

$$\left|E \cap \binom{\mathcal{A}(G)}{r}\right| \le c_{\epsilon,r}\Big(\log_b(np)\Big)^{\frac{r}{r-1}}.$$

▶ Remark 12. Due to r-uniformity, the amount of future information required to surpass the computational threshold is $O\left(\log^{\frac{r}{r-1}} n\right)$. For $r = 2$, this is the same scaling arising in [32, 19].

β-Independent Sets. For an integer $1 \le \beta \le r-1$, a β-independent set in a hypergraph $H = (V,E)$ is a set $I \subseteq V$ such that $|e \cap I| \le \beta$ for each $e \in E(H)$. These structures interpolate between the notions of *strong* and *weak* independent sets – the cases $\beta = 1$ and $\beta = r-1$ (note that $\beta = r-1$ corresponds to the usual notion of independent sets). While strong and weak independent sets have been heavily studied in the literature (both in the random and deterministic settings [50, 26, 16, 4, 14]), β-independent sets for $1 < \beta < r-1$ are not as well understood.

Regarding $H_r(n,p)$, the statistical threshold was determined by [50] in the sparse regime.[8] A curious feature of these structures is that the online greedy algorithm is not guaranteed to construct a β-independent set. To see this, consider an instance $H \sim H_r(n,p)$ and suppose, for example, at step t we have constructed an independent set I_t thus far. At this point, we do not know the structure of the subhypergraph $H' \coloneqq H[I_t \cup \{v_{t+1}, \dots, v_n\}]$. In particular, while I_t is a β-independent set in $H[\{v_1, \dots, v_t\}]$, it may not be one in H'. Generalizing our results on $H_r(n,p)$ – and, indeed, those of [21] in the sparse regime – to $\beta < r-1$ is an interesting direction.

Distance-k Independent Sets. In a similar flavor, a distance-k independent set in a graph $G=(V,E)$ is a set $I \subseteq V$ such that the vertices in I are pairwise of distance at least $k+1$; equivalently, it is an independent set in G^k (note that one recovers the usual notion of independent sets for $k=1$). As above, the greedy algorithm is not guaranteed to construct a distance-k independent set in a graph G.

Very recently, the growth rate of the statistical threshold for the distance-k independence number was determined for $G(n,p)$ [24]. They do not pin down the leading constant factor and raise it as an open problem (see the discussion after [24, Conjecture 8]). In the deterministic setting, there are a number of results for special graph classes, most notably for line graphs [45, 8, 53]. We remark that the problem is trivial for random graphs when $p \geq \sqrt{\frac{2\log n}{n}}$ and $k \geq 2$ as such graphs have diameter 2 [9]. It would be interesting to investigate the problem in the sparse regime.

▶ **Question 13.** *Are there statistical–computational gaps for the maximum distance-k independent set problem when $p = o\left(\sqrt{\frac{2\log n}{n}}\right)$ and $k \geq 2$?*

References

1 Problem list of the AIM Workshop Low-degree polynomial methods in average-case complexity (organizers Sam Hopkins, Tselil Schramm, and Alex Wein), 2024. URL: `https://aimath.org/pastworkshops/lowdegreecomplexityrep.pdf`.

2 Dimitris Achlioptas and Amin Coja-Oghlan. Algorithmic barriers from phase transitions. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 793–802. IEEE, 2008. `doi:10.1109/FOCS.2008.11`.

3 Dimitris Achlioptas and Federico Ricci-Tersenghi. On the solution-space geometry of random constraint satisfaction problems. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 130–139, 2006. `doi:10.1145/1132516.1132537`.

4 Miklós Ajtai, János Komlós, Janos Pintz, Joel Spencer, and Endre Szemerédi. Extremal uncrowded hypergraphs. *Journal of Combinatorial Theory, Series A*, 32(3):321–335, 1982. `doi:10.1016/0097-3165(82)90049-8`.

5 Noga Alon and Joel H Spencer. *The probabilistic method*. John Wiley & Sons, 2016.

6 James Anderson, Anton Bernshteyn, and Abhishek Dhawan. Coloring graphs with forbidden bipartite subgraphs. *Combinatorics, Probability and Computing*, 32(1):45–67, 2023. `doi:10.1017/S0963548322000104`.

7 Afonso S Bandeira, Amelia Perry, and Alexander S Wein. Notes on computational-to-statistical gaps: predictions using statistical physics. *arXiv preprint*, 2018. `arXiv:1803.11132`.

8 Richard Bi, Peter Bradshaw, Abhishek Dhawan, and Jingwei Xu. The strong chromatic index of $k_\{t,t\}$-free graphs. *arXiv preprint*, 2026. `arXiv:2603.15207`.

[8] They actually prove a stronger result in terms of the coloring variant of the parameter.

9 Béla Bollobás. The diameter of random graphs. *Transactions of the American Mathematical Society*, 267(1):41–52, 1981.

10 Béla Bollobás. *Random Graphs*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2 edition, 2001.

11 Béla Bollobás and Paul Erdös. Cliques in random graphs. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 80(3), pages 419–427. Cambridge University Press, 1976.

12 Guy Bresler and Brice Huang. The algorithmic phase transition of random k-sat for low degree polynomials. *arXiv preprint*, 2021. arXiv:2106.02129.

13 Amin Coja-Oghlan and Charilaos Efthymiou. On independent sets in random graphs. *Random Structures & Algorithms*, 47(3):436–486, 2015. doi:10.1002/RSA.20550.

14 Jonathan Cutler and AJ Radcliffe. Hypergraph independent sets. *Combinatorics, Probability and Computing*, 22(1):9–20, 2013. doi:10.1017/S0963548312000454.

15 Ewan Davies, Matthew Jenssen, Will Perkins, and Barnaby Roberts. On the average size of independent sets in triangle-free graphs. *Proceedings of the American Mathematical Society*, 146(1):111–124, 2018.

16 Abhishek Dhawan. Fractional coloring via entropy. https://arxiv.org/abs/2603.17730 (preprint), 2026.

17 Abhishek Dhawan, Nhi U Dinh, Eren C Kızıldağ, Neeladri Maitra, and Bayram A Şahin. Algorithmic phase transition for large independent sets in dense hypergraphs. *arXiv preprint*, 2026. arXiv:2605.05618.

18 Abhishek Dhawan, Oliver Janzer, and Abhishek Methuku. Independent sets and colorings of $k_{t,t,t}$-free graphs. *arXiv preprint*, 2025. arXiv:2511.17191.

19 Abhishek Dhawan, Eren C Kızıldağ, and Neeladri Maitra. Sharp online hardness for large balanced independent sets. *arXiv preprint*, 2025. arXiv:2508.20785.

20 Abhishek Dhawan, Abhishek Methuku, and Minh-Quan Vo. The independence number of uncrowded hypergraphs: bounds matching the shattering threshold. *arXiv preprint*, 2026. arXiv:2606.18048.

21 Abhishek Dhawan and Yuzhou Wang. The low-degree hardness of finding large independent sets in sparse random hypergraphs. *SIAM Journal on Discrete Mathematics*, 40(1):374–419, 2026. doi:10.1137/24M1701575.

22 Uriel Feige. Relations between average case complexity and approximation complexity. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 534–543, 2002. doi:10.1145/509907.509985.

23 Uriel Feige and Shimon Kogan. Hardness of approximation of the balanced complete bipartite subgraph problem. *Dept. Comput. Sci. Appl. Math., Weizmann Inst. Sci., Rehovot, Israel, Tech. Rep. MCS04-04*, 2004.

24 Alan Frieze, Ross Kang, Aditya Raut, Michelle Sweering, and Hilde Verbeek. Coloring powers of random graphs. *arXiv preprint*, 2026. arXiv:2604.14006.

25 Alan Frieze and Colin McDiarmid. Algorithmic theory of random graphs. *Random Structures & Algorithms*, 10(1-2):5–42, 1997. doi:10.1002/(SICI)1098-2418(199701/03)10:1/2\%3C5::AID-RSA2\%3E3.0.CO;2-Z.

26 Alan Frieze and Dhruv Mubayi. Coloring simple hypergraphs. *Journal of Combinatorial Theory, Series B*, 103(6):767–794, 2013. doi:10.1016/J.JCTB.2013.09.003.

27 Alan Frieze and Nicholas C Wormald. Random k-sat: A tight threshold for moderately growing k. *Combinatorica*, 25(3):297–306, 2005. doi:10.1007/S00493-005-0017-3.

28 Alan M Frieze. On the independence number of random graphs. *Discrete Mathematics*, 81(2):171–175, 1990. doi:10.1016/0012-365X(90)90149-C.

29 David Gamarnik. The overlap gap property: A topological barrier to optimizing over random structures. *Proceedings of the National Academy of Sciences*, 118(41), 2021.

30 David Gamarnik. Turing in the shadows of nobel and abel: an algorithmic story behind two recent prizes. *arXiv preprint*, 2025. doi:10.48550/arXiv.2501.15312.

31 David Gamarnik, Aukosh Jagannath, and Alexander S Wein. Low-degree hardness of random optimization problems. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 131–140. IEEE, 2020. doi:10.1109/FOCS46700.2020.00021.

32 David Gamarnik, Eren C Kızıldağ, and Lutz Warnke. Optimal hardness of online algorithms for large independent sets. *arXiv preprint*, 2025. arXiv:2504.11450.

33 David Gamarnik and Quan Li. Finding a large submatrix of a gaussian random matrix. *The Annals of Statistics*, 46(6A):2511–2561, 2018.

34 David Gamarnik, Cristopher Moore, and Lenka Zdeborová. Disordered systems insights on computational hardness. *Journal of Statistical Mechanics: Theory and Experiment*, 2022(11):114015, 2022.

35 David Gamarnik and Madhu Sudan. Limits of local algorithms over sparse random graphs. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 369–376, 2014. doi:10.1145/2554797.2554831.

36 David Gamarnik and Madhu Sudan. Limits of local algorithms over sparse random graphs. *Ann. Probab.*, 45(4):2353–2376, July 2017. doi:10.1214/16-AOP1114.

37 G. R. Grimmett and C. J. H. McDiarmid. On colouring random graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, 77(2):313–324, 1975. doi:10.1017/S0305004100051124.

38 Johan Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182(1):105–142, 1999.

39 Hamed Hatami, László Lovász, and Balázs Szegedy. Limits of locally–globally convergent graph sequences. *Geometric and Functional Analysis*, 24(1):269–296, 2014.

40 Samuel Brink Klevit Hopkins. Statistical inference and the sum of squares method, 2018.

41 Brice Huang and Mark Sellke. Tight lipschitz hardness for optimizing mean field spin glasses. *arXiv preprint*, 2021. arXiv:2110.07847.

42 Brice Huang and Mark Sellke. Algorithmic threshold for multi-species spherical spin glasses. *arXiv preprint*, 2023. doi:10.48550/arXiv.2303.12172.

43 Brice Huang and Mark Sellke. Strong low degree hardness for stable local optima in spin glasses. *arXiv preprint*, 2025. doi:10.48550/arXiv.2501.06427.

44 Mark Jerrum. Large cliques elude the metropolis process. *Random Structures & Algorithms*, 3(4):347–359, 1992. doi:10.1002/RSA.3240030402.

45 Tomáš Kaiser and Ross J Kang. The distance-t chromatic index of graphs. *Combinatorics, Probability and Computing*, 23(1):90–101, 2014. doi:10.1017/S0963548313000473.

46 Richard M. Karp. The probabilistic analysis of some combinatorial search algorithms, 1976.

47 Subhash Khot. Improved inapproximability results for maxclique, chromatic number and approximate graph coloring. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pages 600–609. IEEE, 2001. doi:10.1109/SFCS.2001.959936.

48 Jeong Han Kim. On Brooks' theorem for sparse graphs. *Combinatorics, Probability and Computing*, 4(2):97–132, 1995. doi:10.1017/S0963548300001528.

49 Michael Krivelevich. Coloring random graphs—an algorithmic perspective. In *Mathematics and Computer Science II: Algorithms, Trees, Combinatorics and Probabilities*, pages 175–195. Springer, 2002.

50 Michael Krivelevich and Benny Sudakov. The chromatic numbers of random hypergraphs. *Random Structures & Algorithms*, 12(4):381–403, 1998. doi:10.1002/(SICI)1098-2418(199807)12:4\%3C381::AID-RSA5\%3E3.0.CO;2-P.

51 Dmitriy Kunisky, Alexander S Wein, and Afonso S Bandeira. Notes on computational hardness of hypothesis testing: Predictions using the low-degree likelihood ratio. In *ISAAC Congress (International Society for Analysis, its Applications and Computation)*, pages 1–50. Springer, 2019.

52 Joseph Lauer and Nicholas Wormald. Large independent sets in regular graphs of large girth. *Journal of Combinatorial Theory, Series B*, 97(6):999–1009, 2007. doi:10.1016/J.JCTB.2007.02.006.

53 Mohammad Mahdian. The strong chromatic index of c4-free graphs. *Random Structures & Algorithms*, 17(3-4):357–375, 2000. doi:10.1002/1098-2418(200010/12)17:3/4\%3C357::AID-RSA9\%3E3.0.CO;2-Y.

54 David W Matula. On the complete subgraphs of a random graphproc. In *Second Chapel Hill Conf. on Combinatorial Mathematics and its Applications (Univ. North Carolina, Chapel Hill, NC, 1970)*, pages 356–369, 1970.

55 David W Matula. *The largest clique size in a random graph*. Department of Computer Science, Southern Methodist University Dallas, TX, 1976.

56 Marc Mezard. Glassy phases: a possible origin of computational hardness. In *APS March Meeting Abstracts*, volume 2010, pages A6–004, 2010.

57 Marc Mézard, Thierry Mora, and Riccardo Zecchina. Clustering of solutions in the random satisfiability problem. *Physical Review Letters*, 94(19):197205, 2005.

58 Marc Mézard and Giorgio Parisi. Mean-field theory of randomly frustrated systems with finite connectivity. *Europhysics Letters*, 3(10):1067, 1987.

59 Will Perkins and Yuzhou Wang. On the hardness of finding balanced independent sets in random bipartite graphs. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2376–2397. SIAM, 2024. doi:10.1137/1.9781611977912.85.

60 Mustazee Rahman and Balint Virag. Local algorithms for independent sets are half-optimal. *The Annals of Probability*, 45(3):1543–1577, 2017.

61 James B Shearer. A note on the independence number of triangle-free graphs. *Discrete Mathematics*, 46(1):83–87, 1983. doi:10.1016/0012-365X(83)90273-X.

62 Alexander S Wein. Optimal low-degree hardness of maximum independent set. *Mathematical Statistics and Learning*, 4(3):221–251, 2022.

63 Alexander S Wein. Computational complexity of statistics: New insights from low-degree polynomials. *arXiv preprint*, 2025. doi:10.48550/arXiv.2506.10748.

QMA Lower Bounds for Batch Verification via Approximate Degree

Mark Bun
Boston University, MA, USA

Mandar Juvekar
Boston University, MA, USA

Samuel King
Georgetown University, Washington, DC, USA

Abstract

We study batch verification in QMA query and communication complexity, where the goal is to understand how the resources needed to verify m copies of a Boolean function f depend on m. We give a general technique for proving lower bounds on the witness-query tradeoff needed to batch verify a function f in terms of its approximate degree. Applying this technique to an explicit family of DNF formulas f, we show that attempting to save even a constant factor on the witness length of the baseline approach to batch verifying f necessitates a large polynomial increase in the query cost. We also obtain new lower bounds on the QMA query complexity of read-once CNF formulas and on the surjectivity and k-element distinctness functions. Our lower bounds also lift to give communication analogs of these results.

2012 ACM Subject Classification Theory of computation → Quantum query complexity

Keywords and phrases QMA, approximate degree, query complexity, communication complexity, quantum computation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.69

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2607.08888` [8]

Funding *Mark Bun*: supported in part by NSF award CNS-2046425 and a Sloan Research Fellowship. *Mandar Juvekar*: supported in part by NSF award CNS-2046425 and an IBM PhD Fellowship. *Samuel King*: supported in part by NSF CAREER award CCF-2338730.

Acknowledgements We would like to thank Nadezhda Voronova for her insightful comments on a draft of this paper, Sasha Golovnev for invaluable discussions during this work, and Robin Kothari and Justin Thaler for stimulating our interest in these problems. We would also like to thank Robin Kothari for spotting an error in a previous manuscript.

1 Introduction

A powerful but untrusted server wants to convince a computationally constrained client that it has correctly performed a large list of computations that were delegated to it. It can do so by proving to the client the veracity of m statements $x_1, \ldots, x_m$. As communication is costly, the proof must be as short as possible. If the statements are instances of an NP (or more generally for this work, QMA) language, a baseline strategy is to send as a proof witnesses $w_1, \ldots, w_m$ for all of the statements – the client can then verify each statement individually. Both the communication and the client's computational cost in this strategy are m times the cost for a single statement. For what statements is it possible to do better? What tradeoffs are achievable between communication and computation?


© Mark Bun, Mandar Juvekar, and Samuel King;
licensed under Creative Commons License CC-BY 4.0

This *batch verification* problem has generated exciting research in complexity theory and cryptography. Highly communication-efficient interactive proofs are known for batch verifying UP statements (i.e., NP statements with unique witnesses) [34, 35, 36], while a connection between batch verification and statistical witness indistinguishability [6] suggests extending such a result to all of NP may be impossible. Zero-knowledge batch verification protocols are known for problems admitting non-interactive statistical zero-knowledge proofs [25, 26, 31, 27]. Under cryptographic assumptions, computationally sound noninteractive batch arguments are known for all of NP [23, 16, 15, 18, 42, 22]; recent work has even constructed succinct interactive (classical) batch arguments for QMA [20]. In cryptography, the study of batch verification of NP statements has led to new constructions of succinct non-interactive arguments (SNARGs) [15, 24] and non-interactive zero-knowledge arguments (NIZKs) [14] for more expressive classes and from weaker assumptions. Finally, [32] recently initiated a study of quantum witness indistinguishable proofs and showed that they are implied by quantum batch proofs.

In this work, we study the batch verification problem for quantum Merlin-Arthur (QMA) proofs in the query and communication models. These models abstract computation to settings where techniques are available for proving strong lower bounds, yet which shed both conceptual and technical light on the corresponding Turing machine model. In the QMA *query* or *black-box* model, hereafter denoted by $\mathrm{QMA}^{\mathrm{dt}}$, a quantum verifier (the resource-bounded mortal "Arthur") wishes to evaluate a known function $f : \{-1,1\}^n \to \{-1,1\}$ on an input $x \in \{-1,1\}^n$.[1] He receives a witness consisting of w qubits from a prover (the all-powerful wizard "Merlin"), and is allowed to make q queries to x in quantum superposition. The protocol is correct if (1) whenever $f(x) = -1$ (i.e., x is a "true" input), there exists a witness that causes the verifier to accept with probability at least $2/3$, and (2) whenever $f(x) = 1$ (i.e., x is a "false" input), every witness causes the verifier to reject with probability at least $2/3$. The cost of such a protocol is the sum $w + q$ of the witness length and the number of verifier queries, though we are also interested in understanding the tradeoff between these parameters. The QMA query complexity of a Boolean function f, denoted $\mathsf{QMA}^{\mathsf{dt}}(f)$, is the least cost of a protocol computing f.

The batch verification problem for QMA query complexity can thus be stated as follows: under what conditions on f does the function $(\mathsf{AND}_m \circ f^m)(x_1, \ldots, x_m) := f(x_1) \wedge \cdots \wedge f(x_m)$ have low $\mathrm{QMA}^{\mathrm{dt}}$ complexity? But what should "low" mean here? In the quantum setting, Grover's search algorithm gives a way to "batch compute" all functions with bounded-error algorithms: when f has a bounded-error quantum query protocol (i.e., one that does not make use of a witness) with query cost q, Grover search implies that $\mathsf{AND}_m \circ f^m$ has a bounded-error quantum query protocol with cost only $O(\sqrt{m} \cdot q)$ [21]. This methodology extends to $\mathrm{QMA}^{\mathrm{dt}}$ protocols. If f admits a $\mathrm{QMA}^{\mathrm{dt}}$ protocol with witness length w_f and query complexity q, one can compute $\mathsf{AND}_m \circ f^m$ by taking as witness all m individual witnesses (for a total witness length of $w = m \cdot w_f$) and using Grover search to verify all m copies using $O(\sqrt{m} \cdot q)$ queries. Our batch verification problem thus becomes: for what functions f can one beat this baseline protocol for $\mathsf{AND}_m \circ f^m$ that uses Grover search to verify all m individual witnesses?

Our first main result establishes a strong limitation on batching even when f is highly structured and admits an efficient NP query protocol.

[1] The notation $\mathrm{QMA}^{\mathrm{dt}}$ alludes to yet another view of query algorithms as generalizations of decision trees. Here and throughout the paper we identify -1 with "true" and 1 with "false."

▶ **Theorem 1** (Informal). *For every $\delta > 0$ there is an explicitly given polynomial-size, constant-width DNF $f : \{-1,1\}^n \to \{-1,1\}$ such that every $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$, $m = n^{O(1)}$, with witness length $w \le cm \log n$ for a suitable constant $c \in (0,1)$ must make at least $\widetilde{\Omega}(n^{1-\delta}\sqrt{m/w})$ queries.*

One can interpret this result as follows. Since the function f is computed by a polynomial-size, constant-width DNF, it admits an NP query protocol with witness length $O(\log n)$ and $O(1)$ queries. The baseline strategy for verifying $\mathsf{AND}_m \circ f^m$ thus entails a proof with witness length $O(m \log n)$ and uses $O(\sqrt{m})$ queries. Theorem 1 shows that if one seeks to beat this baseline using a protocol with witness length just a constant factor smaller than this $O(m \log n)$, then the requisite query complexity jumps to $\Omega(n^{1-\delta})$, which is $\Omega(m)$ whenever $m = O(n^{1-\delta})$.

We prove Theorem 1 by proving a more general connection between the $\mathrm{QMA}^{\mathrm{dt}}$ complexity of batch verifying f with the *approximate degree* of f. The ε-approximate degree of f, denoted $\mathrm{adeg}_\varepsilon(f)$, is the minimum degree of a real-valued polynomial that approximates f point-wise to additive error ε. Approximate degree is an extensively studied measure, and has many connections to randomized and quantum query and communication complexity (see, e.g., [13]). In particular, Beals et al. [5] showed that the acceptance probability of every q-query quantum algorithm can be written as a degree-$2q$ polynomial in its input, and hence the ε-error quantum query complexity of a function f is always at least half of its ε-approximate degree.

We show that lower bounds on the ε-approximate degree of f for various settings of ε yield witness-query tradeoffs for $\mathrm{QMA}^{\mathrm{dt}}$ batch verifying f.

▶ **Theorem 2** (Informal). *Let $f : \{-1,1\}^n \to \{-1,1\}$ be a function.*

1. *If $\mathrm{adeg}_{1-1/m}(f) > d$ then every $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length $w = O(m)$ requires $\Omega(d\sqrt{m/w})$ queries.*
2. *If $\mathrm{adeg}_{1/3}(f) > d$ then every $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length at most m requires $\Omega(d)$ queries.*

As we discuss below, previous work derived $\mathrm{QMA}^{\mathrm{dt}}$ lower bounds from more stringent variants of approximate degree lower bounds. However, to our knowledge, ours is the first to draw a direct connection between $\mathrm{QMA}^{\mathrm{dt}}$ complexity and the standard notion of approximate degree.

As an illustrative example, let f be the complement of the element distinctness function on n bits (defined in Section 2.1), which is known to have approximate degree $\tilde{\Omega}(n^{2/3})$ [3, 4]. At the same time, f admits a $\mathrm{QMA}^{\mathrm{dt}}$ protocol with witness length $O(\log n)$ and query complexity $O(\sqrt{\log n})$: the witness consists of the location of a collision, and the verifier uses Grover search to check the collision. So the naïve batching strategy achieves witness length $O(m \log n)$ and query complexity $O(\sqrt{m \log n})$. Item 2 of Theorem 2 shows that any protocol improving on this witness length by even an $O(\log n)$ factor must have its query complexity jump to $\widetilde{\Omega}(n^{2/3})$, which is significantly larger for any $m \ll n^{4/3}$. Item 1 of the theorem gives more fine-grained tradeoffs when one can obtain lower bounds on the approximate degree of f to *large* error. Similar tradeoffs apply to other functions that admit efficient $\mathrm{QMA}^{\mathrm{dt}}$ protocols such as DNFs and the complement of the surjectivity function.

Beyond the intrinsic interest of the batch verification problem, Theorem 2 also serves as a valuable technical tool for understanding the $\mathrm{QMA}^{\mathrm{dt}}$ complexity of specific functions that have been central to the study of quantum query complexity. Sherstov and Thaler [41] used $\mathrm{QMA}^{\mathrm{dt}}$ lower bounds on functions of the form $\mathsf{AND}_m \circ f^m$ to give the sharpest known lower bounds for the influential element distinctness and permutation testing problems. For

technical reasons elaborated on below, their method falls short of applying to closely related functions, such as the two-level AND-OR tree, the k-element distinctness function, and the surjectivity function. Our framework for proving batch verification lower bounds bypasses the limitations from prior work, allowing us to prove the following lower bounds for these functions.

▶ **Theorem 3.** *Let* SURJ_n *denote the surjectivity function on* n *bits and* $k\mathsf{ED}_n$ *the* k*-element distinctness function on* n *bits.*

1. $\mathsf{QMA}^{\mathsf{dt}}(\mathsf{AND}_{n^{1/3}} \circ \mathsf{OR}_{n^{2/3}}^{n^{1/3}}) = \Omega(n^{1/3})$.
2. $\mathsf{QMA}^{\mathsf{dt}}(\mathsf{SURJ}_n) = \widetilde{\Omega}(n^{3/7})$.
3. $\mathsf{QMA}^{\mathsf{dt}}(k\mathsf{ED}_n) = \widetilde{\Omega}(n^{\frac{3k-1}{7k-1}})$.

The best lower bounds that followed straightforwardly from prior work were $\mathsf{QMA}^{\mathsf{dt}}(\mathsf{AND}_{n^{1/3}} \circ \mathsf{OR}_{n^{2/3}}^{n^{1/3}}) = \Omega(n^{1/6})$, $\mathsf{QMA}^{\mathsf{dt}}(\mathsf{SURJ}_n) = \widetilde{\Omega}(n^{1/4})$ and $\mathsf{QMA}^{\mathsf{dt}}(k\mathsf{ED}_n) \geq \mathsf{QMA}^{\mathsf{dt}}(\mathsf{ED}_n) = \widetilde{\Omega}(n^{2/5})$.

1.1 Technical Overview

Our main results follow from a general result (Corollaries 12 and 13, which formalize Theorem 2) showing that $\mathsf{AND}_m \circ f^m$ has high $\mathrm{QMA}^{\mathrm{dt}}$ complexity whenever f has high *approximate degree*.

Suppose there is a QMA protocol for a function F using witness length w and q queries. The in-place amplification technique of Marriott and Watrous [30] gives rise to a protocol for F with improved error 2^{-8w} using witness length w and $O(qw)$ queries. Replacing the witness in this protocol with the maximally mixed state yields a witness-free quantum query protocol for F, and hence an approximating polynomial p with the following properties:

$$F(x) = -1 \implies -2^{8w} \leq p(x) \leq -1,$$
$$F(x) = 1 \implies 1 - 2^{-8w} \leq p(x) \leq 1.$$

That is, p very tightly approximates F on its false inputs, but (roughly) subject to having the right sign behavior, can become exponentially large on true inputs. Thus, in order to prove a lower bound on the QMA query complexity of F, it suffices to rule out low-degree approximating polynomials p of this form. Up to this point, this lower bound method, which is inspired by the complexity class containment $\mathsf{QMA} \subseteq \mathsf{SBQP}$, is well-established for proving QMA query complexity lower bounds [1, 2, 41, 17]. We refer to such an approximating polynomial p as an "SBQP approximation" to F.

Prior work proved degree lower bounds for SBQP approximations by relating it to other better-studied notions of polynomial approximation, including ε-approximate degree in the "large error" regime where $\varepsilon \approx 1 - 2^{-w}$, threshold degree, and vanishing error one-sided approximate degree. We depart from prior work by studying SBQP approximations more directly. We do this using the method of dual polynomials, whereby one proves approximability lower bounds by constructing dual solutions to a linear program capturing the approximation problem. We introduce a notion of "relaxed SBQP duals" to capture feasible solutions in this dual space that are relatively clean to construct and reason about. A relaxed SBQP dual for a function F is a real-valued function Φ that is ε-correlated with F for some $\varepsilon \geq 2^{-8w}$, orthogonal to all low degree polynomials, and which has "almost one-sided error" in the sense that the total weight it places on $\{x : F(x) = -1 \wedge \Phi(x) > 0\}$ is at most $2^{-8w}\varepsilon$. The proof of our general theorem proceeds by showing that whenever f has high approximate degree, we can construct an appropriate relaxed SBQP dual for the function $\mathsf{AND}_m \circ f^m$. The connection outlined above, in turn, implies our $\mathrm{QMA}^{\mathrm{dt}}$ tradeoffs.

To construct this relaxed SBQP dual, we appeal to the fact that the high approximate degrees of AND_m and f are themselves witnessed by dual objects ψ and φ, respectively. Building on a construction of Sherstov [38], we show how to combine ψ and φ to produce a new dual witness to the high SBQP degree of the composed function $F = \mathsf{AND}_m \circ f^m$. Note that Sherstov's original construction was tailored to showing only that F has high approximate degree. Since we need a stronger SBQP degree lower bound, our analysis needs to first take advantage of the fact that ψ actually witnesses the high *one-sided* approximate degree of AND_m. That is, it rules out one-sided approximations which tightly approximate AND_m on false inputs, but may be arbitrarily unbounded on true inputs. This manifests in the dual witness ψ as the property that it places zero weight on the set $\{x : \mathsf{AND}_m(x) = -1 \wedge \psi(x) > 0\}$. Second, our analysis needs to keep tighter track of how errors on the "true" side of F accumulate to show that it has the required almost one-sided error.

With Theorem 2 established, Theorem 1 follows by taking the f to be an explicit, constant-width, polynomial-size DNF with approximate degree $\Omega(n^{1-\delta})$ as constructed by Sherstov [40]. We obtain the results in Theorem 3 by building on the ideas of Sherstov and Thaler [41]. For every parameter m, the functions $k\mathsf{ED}_n$ and SURJ_n contain, as subfunctions, $\mathsf{AND}_m \circ (k\mathsf{ED}_{n/m})^m$ and $\mathsf{AND}_m \circ \mathsf{SURJ}^m_{n/m}$, respectively. We can thus invoke our general theorem on these subfunctions for appropriate choices of m that balance the final witness length and query costs in our lower bound.

The reason why Sherstov and Thaler could use this insight to prove lower bounds for element distinctness and permutation testing, but not for k-element distinctness or surjectivity, owes to a gap between the type of polynomial approximation lower bounds they could prove on functions of the form $\mathsf{AND}_m \circ f^m$ and the type they identified as yielding SBQP approximation lower bounds. Specifically, they showed a lower bound on the vanishing-error approximate degree $\widetilde{\deg}_\varepsilon(\mathsf{AND}_m \circ f^m)$, but needed to invoke a stronger lower bound on the *one-sided* vanishing-error approximate degree $\widetilde{\deg}^-_\varepsilon(\mathsf{AND}_m \circ f^m)$ to rule out SBQP approximations. For some functions with special structure, including AND, element distinctness, and permutation testing, ordinary approximate degree and one-sided approximate degree fortuitously coincide, and hence vanishing-error approximate degree lower bounds imply $\mathrm{QMA}^{\mathrm{dt}}$ lower bounds. But such a coincidence is not (yet) known for k-distinctness or surjectivity, and false for other simple functions such as the two-level AND-OR tree. By working directly with SBQP approximations via our relaxed SBQP duals, we can prove $\mathrm{QMA}^{\mathrm{dt}}$ lower bounds for arbitrary composed functions $\mathsf{AND}_m \circ f^m$, regardless of whether ordinary and one-sided approximate degree agree.

1.2 Lifting to QMA Communication Lower Bounds

In the QMA *communication* model, denoted by $\mathrm{QMA}^{\mathrm{cc}}$, Alice holds an input $x \in \{-1,1\}^n$, Bob holds an input $y \in \{-1,1\}^n$, and they wish to compute a joint function $f : \{-1,1\}^n \times \{-1,1\}^n \to \{-1,1\}$ of their inputs. Bob receives a witness of w qubits from the prover and the parties have unlimited access to shared entanglement. They wish to minimize the amount of communication between them (including the witness) required to compute $f(x,y)$. The success criterion, cost of a protocol, and $\mathrm{QMA}^{\mathrm{cc}}$ complexity of a function f are defined analogously to the query model.

Sherstov's pattern matrix method [37] shows how to generically "lift" an approximate degree lower bound for a function f to a communication lower bound for a related function $G := f \circ g^n$, where g is a certain constant-sized two-party gadget. By adapting the pattern matrix method to show how to lift SBQP degree lower bounds to QMA communication lower

bounds, we establish QMA lower bounds for communication analogs of the batch verification problems we study. Our analysis builds closely on prior generalizations of the pattern matrix method lifting one-sided approximate degree to (multiparty) MA communication [19, 40] and to QMA communication [28], but the formulation of the statement that is most useful to us appears to be new. We defer these results to the full version of this paper [8].

1.3 Open Questions

- Is the witness-query tradeoff given by Theorem 2 tight? That is, can one find a function f such that for every $w = O(m)$, $\mathsf{AND}_m \circ f^m$ admits a $\mathrm{QMA}^{\mathrm{dt}}$ protocol with witness length w and query complexity $O(\mathrm{adeg}_{1-1/m}(f) \cdot \sqrt{m/w})$?
- Our $\mathrm{QMA}^{\mathrm{dt}}$ batch verification lower bounds depend on approximate degree lower bounds for the base function f. Recall that approximate degree is a lower bound on the quantum query complexity of f, but can be much smaller. Can one obtain similar batch verification lower bounds starting from arbitrary quantum query lower bounds on f?
- How tight are the $\mathrm{QMA}^{\mathrm{dt}}$ lower bounds in Theorem 3? Can our ideas be used to obtain improved $\mathrm{QMA}^{\mathrm{dt}}$ lower bounds for element distinctness and permutation testing?

2 Preliminaries

We let $\mathbb{N} = \{1, 2, \ldots\}$ denote the set of natural numbers and $\mathbb{R}$ denote the set of real numbers. We write $[n]$ for the set $\{1, 2, \ldots, n\}$.

We identify the Boolean values "true" and "false" with -1 and 1 respectively. The functions $\mathsf{OR}_n : \{-1,1\}^n \to \{-1,1\}$ and $\mathsf{AND}_n : \{-1,1\}^n \to \{-1,1\}$ are the usual functions $\mathsf{OR}_n(x) = \bigvee_{i=1}^n x_i$ and $\mathsf{AND}_n(x) = \bigwedge_{i=1}^n x_i$. The Hamming weight of a string $x \in \{-1,1\}^n$, denoted $|x|$, is the number of indices i such that $x_i = -1$. A function f is said to be computable by a *disjunctive normal form (DNF)* formula of size s and width w if there exist $S_1, S_2, \ldots, S_s \subseteq [n]$ with cardinality w such that $f(x) = \bigvee_{i \in [s]} \bigwedge_{j \in S_i} x_j$. The conjunctions $\bigwedge_{j \in S_i} x_j$ are called *terms*.

We use the following standard real-valued functions. We use $\exp_k$ to denote the base-k exponential, i.e., $\exp_k(x) = k^x$. The base-2 logarithm is denoted by $\log(\cdot)$. For any predicate P, $\mathbb{1}[P(x)]$ equals 1 if $P(x)$ is true and 0 otherwise. The notation $|x|$ represents the absolute value, Hamming weight, or cardinality of x depending on if x is a real number, binary string, or set respectively.

For real valued functions f and g with the same finite support X, we use the usual notation for the ℓ_1 norm and inner product.

$$\|f\|_1 = \sum_{x \in X} |f(x)|$$
$$\langle f, g \rangle = \sum_{x \in X} f(x) g(x)$$

We often refer to the inner product $\langle f, g \rangle$ as the *correlation* between f and g. The function f is *balanced* if $\sum_{x \in X} f(x) = 0$.

We use superscripts to denote the direct product of a function with itself: for every $f : X \to Y$ and $m \in \mathbb{N}$, $f^m : X^m \to Y^m$ is the function $f^m(x_1, \ldots, x_m) = (f(x_1), \ldots, f(x_m))$. The symbol $\circ$ is used for function composition, so that $(f \circ g)(x) = f(g(x))$. Consequently, for every $g : X \to Y$ and $f : Y^m \to Z$, $f \circ g^m$ denotes the "block composition" of f and g: $(f \circ g^m)(x_1, \ldots, x_m) = f(g(x_1), \ldots, g(x_m))$.

For any set S, we use 2^S to denote the set of all subsets of S. For every $T \subseteq [n]$, we denote by $\chi_T : \{-1,1\}^n \to \{-1,1\}$ the Fourier character $\chi_T(x) = \prod_{i \in T} x_i$. We recall that for every function $f : \{-1,1\}^n \to \mathbb{R}$ there exists a unique function $\widehat{f} : 2^{[n]} \to \mathbb{R}$ called the *Fourier transform* of f such that for all x,

$$f(x) = \sum_{T \subseteq [n]} \widehat{f}(T)\chi_T(x).$$

Furthermore, this unique function is given by

$$\widehat{f}(T) = 2^{-n} \sum_{x \in \{-1,1\}^n} f(x)\chi_T(x).$$

2.1 Polynomial approximation

We say that a polynomial p *ε-approximates* a function $f : \{-1,1\}^n \to \{-1,1\}$ if for all $x \in \{-1,1\}^n$, $|f(x) - p(x)| \leq \varepsilon$. The *$\varepsilon$-approximate degree*, denoted $\mathrm{adeg}_\varepsilon(f)$, of f is the minimum degree of a polynomial that ε-approximates f. A polynomial p *ε-one-sided approximates* f if the following conditions hold.

1. For every $x \in f^{-1}(-1)$, we have $p(x) \in (-\infty, -1 + \varepsilon]$.
2. For every $x \in f^{-1}(1)$, we have $p(x) \in [1 - \varepsilon, 1 + \varepsilon]$.

The *ε-one-sided approximate degree* of f, denoted $\mathrm{adeg}^-_\varepsilon(f)$, is the minimum degree of a polynomial that ε-one-sided approximates f. One can also define a complementary notion of one-sided approximation where the approximating polynomial must tightly approximate true inputs and is allowed to be arbitrarily positive on false inputs, though we do not consider that notion or its associated one-sided approximate degree measure in this paper. For both one-sided and regular approximate degree, when ε is not specified, we assume that $\varepsilon = 1/3$.

We recall the standard dual formulation of approximate degree.

▶ **Theorem 4** ([37, 11]). *Let $f : \{-1,1\}^n \to \{-1,1\}$ be a Boolean function. Let $d \geq 1$ be an integer and $\varepsilon > 0$ be a real number. Then $\mathrm{adeg}_\varepsilon(f) > d$ if and only if there exists a function $\psi : \{-1,1\}^n \to \mathbb{R}$ such that*

1. *$\langle f, \psi \rangle \geq \varepsilon$,*
2. *$\|\psi\|_1 = 1$, and*
3. *for every $T \subseteq [n]$ with $|T| \leq d$, $\langle \psi, \chi_T \rangle = 0$.*

Furthermore, $\mathrm{adeg}^-_\varepsilon(f) > d$ if and only if there exists a function ψ that satisfies the conditions above and has the additional property that for all x such that $f(x) = -1$, $\psi(x) \leq 0$.

The function ψ is called a *dual polynomial* for the lower bound $\mathrm{adeg}_\varepsilon(f) > d$ or $\mathrm{adeg}^-_\varepsilon(f) > d$ (depending on whether it has the additional property). Condition 3 is often referred to as ψ having "pure high degree at least d."

Sherstov showed that there exist functions computable by polynomial-size, constant-width DNF formulas that have near-maximal approximate degree.

▶ **Theorem 5** ([40]). *For all constants $\delta > 0$ and $c \geq 1$, for every $n \in \mathbb{N}$, there is an explicit function $f_n : \{-1,1\}^n \to \{-1,1\}$ computable by a polynomial-size, constant-width DNF such that $\mathrm{adeg}_{1-1/n^c}(f_n) = \Omega(n^{1-\delta})$.*

"Explicit" here means that there is a uniform algorithm that, given n, prints out a description of f_n in time polynomial in n.

The *surjectivity* function $\mathsf{SURJ}_{N,R} : \{-1,1\}^{N \log R} \to \{-1,1\}$ interprets its input as a list of N numbers in $[R]$ and evaluates to true if and only if every element of $[R]$ appears in the list at least once. As a default we let $n = N \log R$ and $R = N/2$. We write SURJ_n for $\mathsf{SURJ}_{N,R}$ with the default setting of parameters. Bun, Kothari, and Thaler [9] showed that the approximate degree of SURJ_n is $\widetilde{\Omega}(n^{3/4})$.

The *element distinctness* problem $\mathsf{ED}_{N,R} : \{-1,1\}^{N \log R} \to \{-1,1\}$ interprets its input as a list of N numbers in $[R]$ and evaluates to true if and only if no element appears more than once. The canonical setting of parameters is $N = R$, as the function is trivially false when $N > R$ and it was shown by Ambainis [4] that the ε-approximate degree of $\mathsf{ED}_{N,R}$ is the same for all $N \leq R$. Aaronson and Shi [3] showed that $\mathrm{adeg}(\mathsf{ED}_n) = \widetilde{\Theta}(n^{2/3})$ when R is large compared to N. The aforementioned result of Ambainis extended their bounds to our setting of parameters.

The *k-element distinctness* problem $k\mathsf{ED}_{N,R} : \{-1,1\}^{N \log R} \to \{-1,1\}$ is a generalization of element distinctness that evaluates to true if and only if no range element appears k or more times. We use the same canonical setting of parameters as for element distinctness: $k\mathsf{ED}_n = k\mathsf{ED}_{N,N}$ where $n = N \log N$. Mande, Thaler, and Zhu [29] showed that $\mathrm{adeg}(k\mathsf{ED}_n) = \widetilde{\Omega}(n^{3/4 - 1/(4k)})$.

Finally, for our lower bound on the $\mathrm{QMA}^{\mathrm{dt}}$ complexity of the two-level AND-OR tree we will need the lower bound $\mathrm{adeg}(\mathsf{AND}_m \circ \mathsf{OR}_n) = \Omega(\sqrt{mn})$, proved by Bun and Thaler [10] and Sherstov [39].

2.2 Quantum Merlin-Arthur protocols

Let $f : \{-1,1\}^n \to \{-1,1\}$ be a function. A *QMA query protocol* for f with witness length w, query complexity q, completeness error ε_1, and soundness error ε_2, is a quantum oracle algorithm A such that

1. For all $x \in f^{-1}(-1)$ there exists an w-qubit "witness" $|\psi\rangle$ such that $\Pr_A[A^x(|\psi\rangle) = -1] \geq 1 - \varepsilon_1$;
2. For all $x \in f^{-1}(1)$, for every w-qubit state $|\psi\rangle$, $\Pr_A[A^x(|\psi\rangle) = -1] \leq \varepsilon_2$; and
3. For all x and w, $A^x(w)$ makes at most q queries to the input x.

The algorithm A is often referred to as a *verifier*. If a protocol has completeness (resp. soundness) error 0, we say that it has perfect correctness (resp. soundness). When the completeness and/or soundness errors are not specified, we set them to the default value 1/3. The QMA query complexity of f, denoted $\mathsf{QMA}^{\mathsf{dt}}(f)$, is the minimum $w + q$ such that there exists a QMA query protocol for f with witness length w and query complexity q.

QMA communication complexity is defined analogously. A QMA communication protocol for a two-party function $f : X \times Y \to \{-1,1\}$ with witness length w, communication cost c, completeness error ε_1, and soundness error ε_2 is a two-party quantum communication protocol (A, B) with access to shared entanglement such that for all inputs (x, y) and w-qubit states $|\psi\rangle$, the execution $[A(x), B(y, |\psi\rangle)]$ involves at most c qubits of communication and satisfies completeness and soundness conditions analogous to the query case. For a more formal definition of this setting we refer the reader to [33]. The QMA communication complexity of f, denoted $\mathsf{QMA}^{\mathsf{cc}}(f)$ is the minimum $w + c$ such that there exists a QMA communication protocol for f with witness length w and communication cost c.

In models with classical witnesses (such as MA or QCMA), one can reduce completeness and soundness errors by repeatedly executing the verifier and outputting the majority output. This does not work for protocols where the witness is a quantum state, since executing the verifier might alter the witness state. Nevertheless, Marriott and Watrous showed that one

can, in fact, reduce the completeness and soundness errors for QMA protocols. The proof of this theorem can be adapted to achieve the same error reduction in the QMA communication setting.

▶ **Theorem 6** ([30]). *Let $f : \{-1,1\}^n \to \{-1,1\}$ be a function. Suppose there exists a $q(n)$-query quantum algorithm Q, numbers $a(n), b(n) \in [0,1]$, and $w(n), p(n) \leq \mathsf{poly}(n)$ such that*

1. *for all $x \in f^{-1}(-1)$ there exists a $w(n)$-qubit state $|\psi\rangle$ such that $\Pr_Q[Q^x(|\psi\rangle)$ accepts$] \geq a(n)$,*
2. *for all $x \in f^{-1}(1)$, for all $w(n)$-qubit states $|\psi\rangle$, $\Pr_Q[Q^x(|\psi\rangle)$ accepts$] \leq b(n)$, and*
3. *$a(n) - b(n) \geq 1/p(n)$.*

Then for every $r(n) \leq \mathsf{poly}(n)$ there exists a $\mathrm{QMA}^{\mathrm{dt}}$ protocol for f with completeness and soundness error $2^{-r(n)}$, witness length $w(n)$, and query complexity $O(q(n) \cdot r(n) \cdot p(n)^2)$.

2.3 Linear algebra

Our results on communication complexity require some standard concepts from matrix analysis which we recall here. Unless otherwise specified, all matrices we consider are over the field of real numbers. We use the notation $[A_{x,y}]_{x\in X, y\in Y}$ to refer to a $|X| \times |Y|$ matrix A whose rows and columns are indexed by elements of X and Y respectively and whose entries are $A_{x,y}$. For any two matrices $[A_{x,y}]_{x\in X,y\in Y}$ and $[B_{x,y}]_{x\in X,y\in Y}$, the inner product of A and B is defined as

$$\langle A, B\rangle = \sum_{x\in X, y\in Y} A_{x,y}B_{x,y}.$$

The singular values of an $m \times n$ matrix A are denoted by $\sigma_1(A) \geq \sigma_2(A) \geq \cdots \geq \sigma_{\min\{m,n\}}(A) \geq 0$. We recall four standard notions of matrix norm.

$$\|A\|_1 = \sum_{x,y} |A_{x,y}| \qquad \text{(The } L_1 \text{ norm)}$$

$$\|A\| = \max_i \sigma_i(A) \qquad \text{(The spectral norm)}$$

$$\|A\|_\Sigma = \sum_i \sigma_i(A) \qquad \text{(The trace norm)}$$

$$\|A\|_F = \sqrt{\sum_i \sigma_i(A)^2} \qquad \text{(The Frobenius norm)}$$

We will use the following well-known identities. For all compatible real matrices A and B,

$$\|AB\|_\Sigma \leq \|A\|_F \|B\|_F, \text{ and}$$
$$\langle A, B\rangle \leq \|A\|_\Sigma \|B\|.$$

3 SBQP Lower Bounds from High Approximate Degree

In this section we show that if a function f has high large-error approximate degree, then $\mathsf{AND}_m \circ f^m$ has high SBQP cost. We start by defining a notion of SBQP cost in terms of polynomial approximations and deriving a corresponding dual notion for it.

▶ **Definition 7.** *A polynomial (M,ε)-SBQP-approximates $f : \{-1,1\}^n \to \{-1,1\}$ if for all $x \in \{-1,1\}^n$,*

1. *if $f(x) = -1$ then $-M - 1 - \varepsilon \leq p(x) \leq -1 + \varepsilon$, and*
2. *if $f(x) = 1$ then $1 - \varepsilon \leq p(x) \leq 1 + \varepsilon$.*

▶ **Lemma 8.** *For all $f : \{-1,1\}^n \to \{-1,1\}$ and $M^*, \varepsilon, d \geq 0$, if there exists a function $\varphi : \{-1,1\}^n \to \mathbb{R}$ with the following properties, then there does not exist a polynomial of degree at most d that (M^*, ε)-SBQP-approximates f.*

1. $\langle \varphi, f \rangle \geq 2\varepsilon$,
2. $\|\varphi\|_1 \leq 1$,
3. *for all $S \subseteq [n]$ such that $|S| \leq d$, $\langle \varphi, \chi_S \rangle = 0$,*
4. $\sum_{x : f(x) = -1, \varphi(x) > 0} \varphi(x) \leq \varepsilon / M^*$.

Proof. Let $f : \{-1,1\}^n \to \{-1,1\}$ be arbitrary. Fix any $\varepsilon \geq 0$. The smallest M such that there exists a degree-d (M, ε)-SBQP-approximation to f is given by the following linear program.

$$\begin{aligned} \min_{\substack{M \\ p \text{ of deg } < d}} \quad & M & \\ \text{s.t.} \quad & p(x) \geq f(x) - \varepsilon & \forall x \in f^{-1}(1) \\ & p(x) + M \geq f(x) - \varepsilon & \forall x \in f^{-1}(-1) \\ & -p(x) \geq -f(x) - \varepsilon & \forall x \in \{-1,1\}^n \\ & M \geq 0 & \end{aligned}$$

The dual of this linear program is the following.

$$\begin{aligned} \max_{\psi^+, \psi^- : \{-1,1\}^n \to \mathbb{R}} \quad & \sum_{x \in \{-1,1\}^n} (\psi^+(x) - \psi^-(x)) f(x) - \varepsilon \sum_{x \in \{-1,1\}^n} (\psi^+(x) + \psi^-(x)) & \\ \text{s.t.} \quad & \left\langle \psi^+ - \psi^-, \chi_S \right\rangle = 0 & \forall S \subseteq [n], |S| \leq d \\ & \sum_{x : f(x) = -1} \psi^+(x) \leq 1 & \\ & \psi^-, \psi^+ \geq 0 & \end{aligned}$$

By a standard transformation (which appears, e.g., in the derivation of the dual formulation of approximate degree), any optimal solution to the dual program can be turned into one where ψ^+ and ψ^- have disjoint support. Thus, we can set $\psi = \psi^+ - \psi^-$ to get the following equivalent dual linear program.

$$\begin{aligned} \max_{\psi : \{-1,1\}^n \to \mathbb{R}} \quad & \langle \psi, f \rangle - \varepsilon \|\psi\|_1 & \\ \text{s.t.} \quad & \langle \psi, \chi_S \rangle = 0 & \forall S \subseteq [n], |S| \leq d \\ & \sum_{x : f(x) = -1, \psi(x) > 0} \psi(x) \leq 1 & \end{aligned}$$

By weak duality, to show that f does not admit an (M^*, ε)-SBQP-approximation it is sufficient to show that the dual program has objective at least M^*. Suppose we have a φ as in the theorem statement. Consider $\psi = M^* \varphi / \varepsilon$. This ψ satisfies the first constraint of the dual since φ has pure high degree at least d. It satisfies the second constraint since $\sum_{x : f(x) = -1, \psi(x) > 0} \psi(x) = \frac{M^*}{\varepsilon} \sum_{x : f(x) = -1, \varphi(x) > 0} \varphi(x) \leq 1$. Finally, the objective attained by ψ is $\frac{M^*}{\varepsilon} \langle \varphi, f \rangle - \varepsilon \cdot \frac{M^*}{\varepsilon} \|\varphi\|_1 \geq 2M^* - M^* = M^*$. ◀

Functions that satisfy the conditions in Lemma 8 are central to our proofs. We call a φ that has those properties with respect to a function f and parameters M, ε, and d a *relaxed (M, ε, d)-SBQP dual to f*. The word "relaxed" refers to the fact that while the existence of

φ is sufficient for a lower bound on the degree of an (M, ε)-SBQP approximation, it might not be necessary. This is because the proof of Lemma 8 loses tightness when it replaces ψ, which is obtained via LP duality, by the weaker (but easier to reason about) object φ. Just as SBQP approximations are natural restrictions of one-sided approximations (allowing large but bounded error on one side), property 4 of the dual is a natural relaxation of the one-sidedness property of adeg^- duals described in Theorem 4. We use relaxed SBQP duals to prove that if a function f has high approximate degree, then all SBQP approximations to $\mathsf{AND}_m \circ f^m$ must have high degree. To do so, we use the following well-known fact about the vanishing-error one-sided approximate degree of the AND function.

▶ **Theorem 9** ([7]). *There exist universal constants δ_{AND} and δ'_{AND} such that for every $\gamma \in [2^{-m}, 1/3]$, $\delta_{\mathsf{AND}}\sqrt{m\log(1/\gamma)} < \mathrm{adeg}^-_\gamma(\mathsf{AND}_m) \leq \delta'_{\mathsf{AND}}\sqrt{m\log(1/\gamma)}$.*

Strictly speaking, [7] only implies $\mathrm{adeg}_\gamma(\mathsf{AND}_m) = \Theta(\sqrt{m\log(1/\gamma)})$, but it is easy to see that one-sidedness is without loss of generality for the AND function. (This observation goes back to Gavinsky and Sherstov [19].) Suppose $\mathrm{adeg}_\gamma(\mathsf{AND}_m) > d \geq 1$, and let ψ be a dual witness to that lower bound guaranteed by Theorem 4. Since ψ has pure high degree at least 1, we get that $0 = \langle \psi, 1\rangle = \psi(-\mathbf{1}_m) + \sum_{z \neq -\mathbf{1}_m} \psi(z)$. That ψ has correlation at least γ with AND_m combined with the fact that $-\mathbf{1}_m$ is the only -1 instance of AND_m gives us $\gamma \leq \langle \psi, \mathsf{AND}_m\rangle = -\psi(-\mathbf{1}_m) + \sum_{z \neq -\mathbf{1}_m} \psi(z)$. Combining the two equations yields $\psi(-\mathbf{1}_m) \leq -\gamma/2 < 0$, which means ψ is a witness for $\mathrm{adeg}^-_\gamma(\mathsf{AND}_m) > d$. The other ingredient in our construction is the following polynomial p_k.

▶ **Lemma 10** ([38]). *Let k be an even number in $[m-1]$, and define $p_k : [-1,1]^m \to \mathbb{R}$ as the unique degree-k multilinear polynomial such that for all $z \in \{-1,1\}^m$,*

$$p_k(z) = \prod_{i=1}^{k}(|z| - i).$$

Then $p_k(z) \geq 0$ for all $z \in [-1,1]^m$ and for every $\eta \in [0,1)$,

$$\mathop{\mathbb{E}}_{z\sim\Pi(\eta)}[p_k(z)] \leq p_k(\mathbf{1}_m)(1-\eta)^m\left\{1 + \frac{\eta^{k+1}}{(1-\eta)^m}\binom{m}{k+1}\right\},$$

where $\Pi(\eta)$ is the distribution on $\{-1,1\}^m$ where each coordinate is -1 with probability η and 1 with probability $1-\eta$.

We now prove our main connection between approximate degree and SBQP degree. Our proof works by way of constructing a relaxed SBQP dual for $\mathsf{AND}_m \circ f^m$ from (1) a dual witness for the vanishing-error one-sided approximate degree of AND, (2) a dual for the assumed approximate degree lower bound on f, and (3) the polynomials p_k. The construction itself is a renormalized special case of the dual constructed in [38, Theorem 6.1]. Consequently, the correlation, ℓ_1 norm, and pure high degree analyses for the dual follow from the corresponding analyses in that work. Additionally, to get an SBQP dual, we need to show that the dual construction does not place too much mass on "true" inputs that it misclassifies. We do so by taking advantage of the fact that the dual for AND is one-sided and keeping tighter track of how errors on misclassified "true" inputs accumulate.

▶ **Theorem 11.** *Let n be a function of m, and let $\{f : \{-1,1\}^{n(m)} \to \{-1,1\}\}_{m\in\mathbb{N}}$ be a family of functions. Let $\gamma \in [2^{-m}, 1/3]$, $\varepsilon > 0$, $M > 0$, and $k \in \mathbb{N}$ be such that k is even and*

$$\varepsilon \leq \frac{\gamma}{2} - \frac{\eta^{k+1}}{(1-\eta)^m}\binom{m}{k+1} \text{ and} \tag{1}$$

$$\frac{\varepsilon}{M} \geq \frac{\eta^{k+1}}{2(1-\eta)^m}\binom{m}{k+1}. \tag{2}$$

If $\mathrm{adeg}_{1-\eta}(f) > d \geq 1$ then for sufficiently large m, every (M,ε)-SBQP approximation to $F = \mathsf{AND}_m \circ f^m$ has degree $\Omega(d \cdot (\delta_{\mathsf{AND}}\sqrt{m\log(1/\gamma)} - k))$.

Proof. Fix n, $\eta \in (0,1]$, $f : \{-1,1\}^n \to \{-1,1\}$, $d \geq 1$, and suppose $\mathrm{adeg}_{1-\eta}(f) > d$. Suppose γ, ε, M, and k are such that equations 1 and 2 hold. By Lemma 8, it suffices to construct a relaxed $(M, \varepsilon, \Omega(d \cdot (\delta_{\mathsf{AND}}\sqrt{m\log(1/\gamma)} - k)))$-SBQP dual Φ to F.

By Theorems 4 and 9 there is a dual witness $\psi : \{-1,1\}^m \to \mathbb{R}$ for $\mathrm{adeg}^-_\gamma(\mathsf{AND}_m) \geq \delta_{\mathsf{AND}}\sqrt{m\log(1/\gamma)}$. Similarly, there is a dual $\varphi : \{-1,1\}^n \to \mathbb{R}$ witnessing $\mathrm{adeg}_{1-\eta}(f)$. Let $\widetilde{\mu}$ be the distribution on $\{-1,1\}^n$ with probability mass function $\widetilde{\mu}(y) = |\varphi(y)|$, and let μ be a distribution consisting of m independent samples from $\widetilde{\mu}$. For every $z \in \{-1,1\}^m$ define μ_z to be μ conditioned on $(\ldots, \operatorname{sgn}\varphi(x_i), \ldots) = z$.

Note that $\mathbb{E}_{y\sim\widetilde{\mu}}[\operatorname{sgn}\varphi(y)] = \sum_{y\in\{-1,1\}^n} |\varphi(y)| \operatorname{sgn}\varphi(y) = \langle\varphi, \chi_\emptyset\rangle$. Since φ has pure high degree $d \geq 1$, $\langle\varphi,\chi_\emptyset\rangle = 0$ and so $\Pr_{y\sim\widetilde{\mu}}[\varphi(y) > 0] = \Pr_{y\sim\widetilde{\mu}}[\varphi(y) < 0]$. In other words, if $y \sim \widetilde{\mu}$, then $\operatorname{sgn}\varphi(y)$ is a uniformly random bit. Consequently, if $x \sim \mu$, then $(\ldots, \operatorname{sgn}\varphi(x_i), \ldots)$ is uniformly distributed over $\{-1,1\}^m$.

Since φ is a dual for $\mathrm{adeg}_{1-\eta}(f) > d$, $\langle\varphi, f\rangle \geq 1-\eta$. Thus $1-\eta \leq \sum_{y\in\{-1,1\}^n} f(y)\varphi(y) = \mathbb{E}_{y\sim\widetilde{\mu}}[f(y)\operatorname{sgn}\varphi(y)]$. Notice that for all y, $\frac{1-f(y)\operatorname{sgn}\varphi(y)}{2}$ is equal to 1 if $\operatorname{sgn}\varphi(y) \neq f(y)$ and 0 otherwise. Thus

$$\Pr_{y\sim\mu}[\operatorname{sgn}\varphi(y) \neq f(y)] = \mathop{\mathbb{E}}_{y\sim\widetilde{\mu}}\left[\frac{1-f(y)\operatorname{sgn}\varphi(y)}{2}\right] \leq \frac{\eta}{2}.$$

Now define

$$\eta^+ = \Pr_{y\sim\mu}[f(y) \neq \operatorname{sgn}\varphi(y) \mid \varphi(y) > 0],$$
$$\eta^- = \Pr_{y\sim\mu}[f(y) \neq \operatorname{sgn}\varphi(y) \mid \varphi(y) < 0],$$

and $\alpha : \{-1,1\}^n \to [-1,1]$ as

$$\alpha(y) = \begin{cases} \frac{1-2\eta+\eta^-}{1-\eta^-} & \text{if } f(y) = \operatorname{sgn}\varphi(y) = -1 \\ \frac{1-2\eta+\eta^+}{1-\eta^+} & \text{if } f(y) = \operatorname{sgn}\varphi(y) = 1 \\ -1 & \text{otherwise.} \end{cases}$$

The range of α is $[-1,1]$ because $\max\{\eta^+,\eta^-\} \leq \eta$, which holds since $1-\eta \leq 2\Pr_{y\sim\widetilde{\mu}}[\operatorname{sgn}\varphi(y) \neq f(y)] = \frac{1}{2}(1-2\eta^+) + \frac{1}{2}(1-2\eta^-)$. For $x \in (\{-1,1\}^n)^m$, we write $\alpha^m(x)$ to mean $(\ldots, \alpha(x_i), \ldots)$. Note that for $z \in \{-1,1\}^m$ and $i \in [m]$, $\mathbb{E}_{x\sim\mu_z}[\alpha(x_i)] = 1-2\eta$.

We are now ready to present the dual. Let $p_k : [-1,1]^m \to [0,\infty)$ be the degree-k multilinear polynomial given by Lemma 10. Our relaxed SBQP dual $\Phi : (\{-1,1\}^n)^m \to \mathbb{R}$ is defined as

$$\Phi(x_1,\ldots,x_m) = \frac{2^m}{p_k(\ldots,1-2\eta,\ldots)} \cdot \psi(\ldots,\operatorname{sgn}\varphi(x_i),\ldots) \cdot p_k(\alpha^m(x)) \prod_{i=1}^m |\varphi(x_i)|.$$

As we now describe, the required correlation, ℓ_1 norm, and pure high degree properties follow from the corresponding claims in [38]. First, [38, Claim 6.3] tells us that

$$\langle F, \Phi \rangle \geq \gamma - \frac{2\eta^{k+1}}{(1-\eta)^m}\binom{m}{k+1} \geq 2\varepsilon,$$

where the last inequality applies Eq. 1. That $\|\Phi\|_1 = 1$ follows from [38, Claim 6.2]. Finally, [38, Eq. 6.7] and the fact that ψ has pure high degree at least $\delta_{\mathsf{AND}}\sqrt{m\log(1/\gamma)}$ implies that Φ has pure high degree $\Omega(d \cdot (\delta_{\mathsf{AND}}\sqrt{m\log(1/\gamma)} - k))$.

It remains to show that Φ does not place too much mass on -1 instances that it "misclassifies." Notice that the only term in the definition of Φ that is not non-negative is $\psi(\ldots, \operatorname{sgn}\varphi(x_i), \ldots)$. For convenience, let $\beta = p_k(\ldots, 1-2\eta, \ldots)$. We can write the error on -1 instances as

$$\begin{aligned}
&\sum_{x:F(x)=-1,\Phi(x)>0} \Phi(x) = \sum_{x\in(\{-1,1\}^n)^m} \Phi(x)\cdot \mathbb{1}[F(x)=-1 \wedge \Phi(x)>0] \\
&= \frac{2^m}{\beta} \sum_{x\in(\{-1,1\}^n)^m} \psi(\ldots,\operatorname{sgn}\varphi(x_i),\ldots)\cdot p_k(\alpha^m(x_i))\cdot \mathbb{1}[F(x)=-1\wedge\Phi(x)>0]\prod_{i=1}^m |\varphi(x_i)| \\
&= \frac{2^m}{\beta} \mathop{\mathbb{E}}_{x\sim\mu}\left[\psi(\ldots,\operatorname{sgn}\varphi(x_i),\ldots)\cdot p_k(\alpha^m(x))\cdot \mathbb{1}[F(x)=-1\wedge\Phi(x)>0]\right] \\
&= \frac{1}{\beta} \sum_{z\in\{-1,1\}^m} \mathop{\mathbb{E}}_{x\sim\mu_z}\left[\psi(z)\cdot p_k(\alpha^m(x))\cdot \mathbb{1}[F(x)=-1\wedge\psi(z)>0]\right] \\
&= \frac{1}{\beta} \sum_{z:\psi(z)>0} \psi(z) \mathop{\mathbb{E}}_{x\sim\mu_z}\left[p_k(\alpha^m(x))\cdot \mathbb{1}[F(x)=-1]\right]. \qquad (3)
\end{aligned}$$

The second-to-last equality holds since $\Pr_{x\sim\mu}[(\ldots,\operatorname{sgn}\varphi(x_i),\ldots)=z] = 2^{-m}$. Fix z. By multilinearity of p_k, $\mathbb{E}_{x\sim\mu_z}[p_k(\alpha^m(x))] = \beta$. Thus,

$$\begin{aligned}
&\left|\mathop{\mathbb{E}}_{x\sim\mu_z}\left[p_k(\alpha^m(x))\cdot\mathbb{1}[F(x)=-1]\right] - \beta\cdot\mathbb{1}[\mathsf{AND}_m(z)=-1]\right| \\
&\qquad = \left|\mathop{\mathbb{E}}_{x\sim\mu_z}\left[p_k(\alpha^m(x))\cdot\mathbb{1}[F(x)=-1] - p_k(\alpha^m(x))\cdot\mathbb{1}[\mathsf{AND}_m(z)=-1]\right]\right| \\
&\qquad \leq \mathop{\mathbb{E}}_{x\sim\mu_z}\left[\,|p_k(\alpha^m(x))|\cdot\left|\mathbb{1}[F(x)=-1] - \mathbb{1}[\mathsf{AND}_m(z)=-1]\right|\right].
\end{aligned}$$

Recall that $F(x) = \mathsf{AND}_m(\ldots, f(x_i), \ldots)$. So $|\mathbb{1}[F(x)=-1] - \mathbb{1}[\mathsf{AND}_m(z)=-1]|$ is 1 when exactly one of $(\ldots, f(x_i), \ldots)$ and z is equal to $-\mathbf{1}_m$ and 0 otherwise. By the non-negativity of p_k on inputs in $[-1,1]^m$, it follows that

$$\begin{aligned}
&\left|\mathop{\mathbb{E}}_{x\sim\mu_z}\left[p_k(\alpha^m(x))\cdot\mathbb{1}[F(x)=-1]\right] - \beta\cdot\mathbb{1}[\mathsf{AND}_m(z)=-1]\right| \\
&\qquad \leq \mathop{\mathbb{E}}_{x\sim\mu_z}\left[p_k(\alpha^m(x))\cdot\mathbb{1}[(\ldots,f(x_i),\ldots)\neq z]\right] \\
&\qquad = \mathop{\mathbb{E}}_{x\sim\mu_z}\left[p_k(\alpha^m(x))\cdot\big(1-\mathbb{1}[(\ldots,f(x_i),\ldots)=z]\big)\right] \\
&\qquad = \beta - \mathop{\mathbb{E}}_{x\sim\mu_z}\left[p_k(\alpha^m(x))\cdot\mathbb{1}[(\ldots,f(x_i),\ldots)=z]\right].
\end{aligned}$$

Now [38, Eq. 6.8] shows that

$$\mathop{\mathbb{E}}_{x\sim\mu_z}\left[p_k(\alpha^m(x))\cdot\mathbb{1}[(\ldots,f(x_i),\ldots)=z]\right] \geq (1-\eta)^m p_k(\mathbf{1}_m),$$

which means

$$\left| \mathop{\mathbb{E}}_{x\sim\mu_z} \left[p_k(\alpha^m(x)) \cdot \mathbb{1}[F(x) = -1] \right] - \beta \cdot \mathbb{1}[\mathsf{AND}_m(z) = -1] \right| \leq \beta - (1-\eta)^m p_k(\mathbf{1}_m).$$

Combining this bound with Eq. 3, we get

$$\begin{aligned} \sum_{x:F(x)=-1,\Phi(x)>0} \Phi(x) &\leq \frac{1}{\beta} \sum_{z:\psi(z)>0} \psi(z)\big(\beta \cdot \mathbb{1}[\mathsf{AND}_m(z) = -1] + \beta - (1-\eta)^m p_k(\mathbf{1}_m)\big) \\ &= \left(\sum_{z:\psi(z)>0} \psi(z) \right) \left(1 - \frac{(1-\eta)^m p_k(\mathbf{1}_m)}{\beta} \right) \\ &= \frac{1}{2} \left(1 - \frac{(1-\eta)^m p_k(\mathbf{1}_m)}{\beta} \right). \end{aligned}$$

The second-to-last equality uses the one-sidedness of ψ: whenever $\mathsf{AND}_m(z) = -1$, $\psi(z) < 0$. The last equality uses the fact that ψ is balanced.

Now let Π be a distribution on $\{-1,1\}^m$ where each coordinate is an independent, uniformly random element of $\{-1,1\}$. Then by multilinearity, $\mathbb{E}_{z\sim\Pi}[p_k(z)] = \beta$. By Lemma 10,

$$\begin{aligned} \frac{(1-\eta)^m p_k(\mathbf{1}_m)}{\beta} &\geq \left(1 + \frac{\eta^{k+1}}{(1-\eta)^m} \binom{m}{k+1} \right)^{-1} \\ &\geq 1 - \frac{\eta^{k+1}}{(1-\eta)^m} \binom{m}{k+1}. \end{aligned}$$

Plugging this back in and using Eq. 2,

$$\sum_{x:F(x)=-1,\Phi(x)>0} \Phi(x) \leq \frac{\eta^{k+1}}{2(1-\eta)^m} \binom{m}{k+1} \leq \frac{\varepsilon}{M}.$$ ◀

The ensuing corollaries apply Theorem 11 with specific parameters to obtain more "user-friendly" connections between the approximate degree of f and the SBQP degree of $\mathsf{AND}_m \circ f^m$.

▶ **Corollary 12.** *Let n be a function of m, and let $\{f : \{-1,1\}^{n(m)} \to \{-1,1\}\}_{m\in\mathbb{N}}$ be a family of functions. Let $c > 0$ be a constant, $d \geq 1$, and $\varepsilon \in [2^{-m-2}, 1/12]$ such that either $c \geq 1$ or $\varepsilon = 2^{-\Omega(m)}$. If $\widetilde{\deg}_{1-\frac{1}{m^c}}(f) > d$, then there exists a constant $\tau > 0$ such that for sufficiently large m, every (M,ε)-SBQP approximation to $F = \mathsf{AND}_m \circ f^m$ with $M \leq \varepsilon 2^{\tau\sqrt{m\log(1/\varepsilon)}\log m}$ must have degree $\Omega(d\sqrt{m\log(1/\varepsilon)})$. The constant τ depends only on c when $c \geq 1$, and on c as well as the constant in the Ω when $c \in (0,1)$ but $\varepsilon = 2^{-\Omega(m)}$.*

Proof. Let n, f, c, d, and ε be as in the corollary statement. Let $\tau > 0$ be a constant to be set later in this proof. We apply Theorem 11 with

$$\begin{aligned} \eta &= \frac{1}{m^c}, \\ \gamma &= 4\varepsilon, \\ M &= \varepsilon 2^{\tau\sqrt{m\log(1/\varepsilon)}\log m}, \text{ and} \\ k &= \text{the least even number greater than } \frac{\delta_{\mathsf{AND}}\sqrt{m\log(1/\gamma)}}{2}. \end{aligned}$$

This will imply that every (M,ε)-SBQP approximation to $\mathsf{AND}_m \circ f^m$ has degree at least $\Omega(d \cdot (\delta_{\mathsf{AND}}\sqrt{m\log(1/\gamma)} - k)) = \Omega(d\sqrt{m\log(1/\varepsilon)})$. The lower bound extends to smaller M immediately since an (M',ε)-SBQP approximation with $M' < M$ is also an (M,ε)-SBQP approximation.

We now get an upper bound on the quantity $\frac{\eta^{k+1}}{(1-\eta)^m}\binom{m}{k+1}$, which will allow us to show that the premises of Theorem 11 are satisfied. First, suppose $c \geq 1$. We have $(1-\eta)^m = ((1-1/m^c)^{m^c})^{m^{1-c}}$. For large m, this is at least $(1/2e)^{m^{1-c}} \geq 1/2e$. By Theorem 9 there is a universal constant C such that $k \geq C\sqrt{m\log(1/\gamma)}$. We have

$$\frac{\eta^{k+1}}{(1-\eta)^m}\binom{m}{k+1} \leq 2e \cdot m^{-c\,(k+1)} \cdot m^{k+1} = 2em^{-(c-1)(k+1)} = 2^{-\Omega(k\log m)}.$$

The last bound holds because $c \geq 1$ and hence $m^{-(c-1)(k+1)} = 2^{-\Omega(k\log m)}$.

Now suppose $c \in (0,1)$ but $\varepsilon = 2^{-\Omega(m)}$. Since $\gamma = 4\varepsilon$, there is a constant $C_1 \in (0,1]$ such that $\gamma = 2^{-C_1 m}$. Also, $k+1 \leq C_2 m$ for an appropriate constant $C_2 \in (0,1)$ that depends on δ_{AND} and C_1. Using Stirling's approximation,

$$\begin{aligned}
\frac{\eta^{k+1}}{(1-\eta)^m}\binom{m}{k+1} &\leq \frac{m^{-c\,(k+1)}}{(1-m^{-c})^m}\left(\frac{me}{k+1}\right)^{k+1} \\
&= \frac{(m^c)^{m-(k+1)}}{(m^c-1)^m}\left(\frac{e}{C_2}\right)^{C_2 m} \\
&= \exp_2\left((1-C_2)cm\log m - m\log(m^c-1) + C_2 m\log(e/C_2)\right) \\
&= 2^{-\Omega(m\log m)} = 2^{-\Omega(k\log m)},
\end{aligned}$$

where the last bound holds since for large m, $(1-C_2)cm\log m < m\log(m^c+1)$.

Thus in either case, it follows from $k \geq \frac{\delta_{\mathsf{AND}}}{2}\sqrt{m\log(1/\gamma)}$ that for sufficiently large m, $\frac{\eta^{k+1}}{(1-\eta)^m}\binom{m}{k+1} \leq \gamma/4$. Thus $\frac{\gamma}{2} - \frac{\eta^{k+1}}{(1-\eta)^m}\binom{m}{k+1} \geq \frac{\gamma}{4} = \varepsilon$, which establishes Eq. 1.

Eq. 2 holds similarly. Since $k \geq \frac{\delta_{\mathsf{AND}}}{2}\sqrt{m\log(1/\gamma)} \geq C_3\sqrt{m\log(1/\varepsilon)}$ (for an appropriate universal constant C_3), there exists τ' such that

$$\frac{\eta^{k+1}}{2(1-\eta)^m}\binom{m}{k+1} \leq 2^{-\tau' C_3\sqrt{m\log(1/\varepsilon)}\log m}.$$

Setting $\tau = \tau' \cdot C_3$, this quantity is at most ε/M. ◀

▶ **Corollary 13.** *Let n be a function of m, and let $\{f : \{-1,1\}^{n(m)} \to \{-1,1\}\}_{m\in\mathbb{N}}$ be a family of functions. If $\mathrm{adeg}_{1/3}(f) > d \geq 1$, then for sufficiently large m, every $(2^{m-1}, 2^{-m-2})$-SBQP approximation to $F = \mathsf{AND}_m \circ f^m$ must have degree $\Omega(dm)$.*

Proof. By standard error reduction techniques (see, e.g., [13, Section 3.4]), it is possible to transform any degree-k 0.99-approximation to f into a degree-$O(k)$ 1/3-approximation to f. Thus it suffices to prove the corollary assuming $\mathrm{adeg}_{0.99}(f) > d \geq 1$; the corollary as stated follows with at the cost of a constant factor in the degree lower bound.

We use the well-known fact that $\mathrm{adeg}^{-}_{2^{-m}}(\mathsf{AND}_m) \geq m$ (note the absence of a constant factor); the dual witness justifying this is simply the normalized parity function on m bits. Let $f : \{-1,1\}^n \to \{-1,1\}$ be such that $\mathrm{adeg}_{0.99}(f) > d \geq 1$. We apply Theorem 11 with

$$\eta = \frac{1}{100},$$
$$\gamma = 2^{-m},$$
$$\varepsilon = 2^{-m-2},$$
$$M = 2^{m-1}, \text{ and}$$
$$k = \frac{m}{2} - 1.^{2}$$

By Stirling's approximation,

$$\begin{aligned}\frac{\eta^{k+1}}{(1-\eta)^m}\binom{m}{k+1} &\le \frac{1}{100^{k+1}} \cdot \frac{100^m}{99^m} \cdot \left(\frac{me}{k+1}\right)^{k+1} \\ &= \frac{100^{m/2}}{99^m} \cdot (2e)^{m/2} && (k+1 = m/2) \\ &= \left(\frac{\sqrt{200e}}{99}\right)^m \le 2^{-2m}. && (\sqrt{200e}/99 \le 1/4)\end{aligned}$$

Thus for large m,

$$\frac{\gamma}{2} - \frac{\eta^{k+1}}{(1-\eta)^m}\binom{m}{k+1} \ge 2^{-m-1} - 2^{-2m} \ge 2^{-m-2} = \varepsilon \text{ and}$$
$$\frac{\eta^{k+1}}{2(1-\eta)^m}\binom{m}{k+1} \le 2^{-2m-1} = \frac{\varepsilon}{M},$$

satisfying the premises of Theorem 11. ◀

4 QMA^{dt} Lower Bounds

In this section, we apply Corollaries 12 and 13 to obtain various QMA^{dt} lower bounds. Our proofs rely on the following connection between QMA^{dt} protocols and SBQP approximations.

▶ **Lemma 14.** *If $f : \{-1,1\}^n \to \{-1,1\}$ has a* QMA^{dt} *protocol with witness length w and query complexity q, then for every $\varepsilon > 0$ there is an $O(q \cdot \max\{w, \log(1/\varepsilon)\})$-degree polynomial that $(2^{w+2}, \varepsilon)$-SBQP approximates f.*

Proof. Suppose f has a QMA^{dt} protocol with witness length w and query complexity q. Fix $\varepsilon > 0$, and let $\nu = \min\{\frac{\varepsilon}{2}, 2^{-2w}\}$. By Theorem 6, there is a QMA^{dt} protocol P for f with completeness and soundness error at most ν, witness length w, and query complexity $O(q \log(1/\nu))$. Let Q be the witness-free protocol that simulates P with the maximally mixed state as the witness and accepts if and only if P outputs -1. By soundness of P, whenever $f(x) = 1$, $\Pr_Q[Q^x \text{ accepts}] \le \nu$. On the other hand, on inputs where $f(x) = -1$, the probability that Q accepts is at least $(1-\nu)2^{-w} \ge (1-2^{-2w})2^{-w} \ge 2^{-w-1}$.

Let $\widetilde{p}$ be the $O(q \log(1/\nu))$-degree polynomial computing the acceptance probability of Q. Define $p = 1 - 2 \cdot 2^{w+1}\widetilde{p}$. The degree of p is $O(q \log(1/\nu)) = O(q \cdot \max\{w, \log(1/\varepsilon)\})$.

1. If $f(x) = -1$ then $2^{w+1}\widetilde{p}(x) \in [1, 2^{w+1}]$ and so $p(x) \in [1 - 2^{w+2}, -1] \subset [-2^{w+2} - 1 - \varepsilon, -1 + \varepsilon]$.
2. If $f(x) = 1$ then $2^{w+1}\widetilde{p}(x) \in [0, \nu]$ and so $p(x) \in [1 - 2\nu, 1] \subset [1 - \varepsilon, 1 + \varepsilon]$. ◀

[2] We assume for simplicity of exposition that k is an even natural number. The case where $m/2 - 1$ is not an even number follows by setting k to the least even number greater than $m/2 - 1$.

4.1 The QMA^{dt} Complexity of Batch Verification

Our first lower bound exhibits a smooth tradeoff between witness length and query complexity of batch verifying functions with high large-error approximate degree.

▶ **Theorem 15.** *Let n be a function of m, and $\{f : \{-1,1\}^{n(m)} \to \{-1,1\}\}_{m\in\mathbb{N}}$ be an arbitrary family of functions. For all sufficiently large m, if $\mathrm{adeg}_{1-1/m}(f) > d \geq 1$ then every $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length $w = O(m)$ requires $\Omega(d\sqrt{m/w})$ queries.*

Proof. Fix $n(m)$ and $f : \{-1,1\}^{n(m)} \to \{-1,1\}$. Assume $\mathrm{adeg}_{1-1/m}(f) > d \geq 1$. Suppose there is a $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length $w = O(m)$ and query complexity q. We assume that $w \geq \log(12)$; the case where $w < \log(12)$ follows immediately with a smaller constant in the Ω. Let $\varepsilon = \max\{2^{-w}, 2^{-m-2}\} \in [2^{-m-2}, 1/12]$. The premises of Corollary 12 are satisfied with $c = 1$. Thus, by Corollary 12, there exists a constant τ such that every (M,ε)-SBQP approximation to $\mathsf{AND}_m \circ f^m$ with $M \leq \varepsilon 2^{\tau\sqrt{m\log(1/\varepsilon)\log m}}$ has degree $\Omega(d\sqrt{m\log(1/\varepsilon)})$.

By Lemma 14, there is a degree-$O(qw)$ polynomial p that $(2^{w+2},\varepsilon)$-SBQP approximates $\mathsf{AND}_m \circ f^m$. Since $\log(1/\varepsilon) = \Omega(w)$, we have $\tau\sqrt{m\log(1/\varepsilon)}\log m = \Omega(\sqrt{mw}\log m)$. Thus for large m, $2^{w+2} \leq \varepsilon 2^{\tau\sqrt{m\log(1/\varepsilon)\log m}}$. So $O(qw) = \deg(p) = \Omega(d\cdot\sqrt{m\log(1/\varepsilon)}) = \Omega(d\sqrt{mw})$. Rearranging yields the desired lower bound. ◀

Observe that the proof of Theorem 15 can actually tolerate witness lengths all the way up to $\tau' m\log m$ for some fixed constant $\tau' \in (0,1)$. Roughly speaking, it shows that whenever $w \leq \tau' m\log m$, any $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length w must have query complexity at least $\Omega(\mathrm{adeg}_{1-1/m}(f)\sqrt{m/w})$. Combining this observation with Theorem 5 we show that even a constant-factor improvement in witness length over the trivial protocol for batch verifying DNFs incurs a large increase in query complexity.

▶ **Corollary 16.** *For all sufficiently large n and all $\delta > 0$ there exists an explicitly given function $f : \{-1,1\}^n \to \{-1,1\}$ computable by a polynomial-size, constant-width DNF and a constant $k \in \mathbb{N}$ such that for all $m = n^{O(1)} \leq n$, the following hold.*

1. *There is a $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length $km\log n$ and query complexity $O(k\sqrt{m})$.*
2. *There is a constant $\tau' \in (0,1)$ such that every $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length at most $\tau' m\log n$ must have query complexity $\min\{\Omega(n^{1-\delta}\sqrt{m/w}), \Omega(n^{1-\delta}m/w)\}$.*

Proof. By Theorem 5 for every n and $\delta > 0$ there exists an explicitly given function $f : \{-1,1\}^n \to \{-1,1\}$ computable by a polynomial-size, constant-width DNF such that $\mathrm{adeg}_{1-1/n}(f) > n^{1-\delta}$. Let $k \in \mathbb{N}$ be a constant such that the DNF computing f has width at most k and size a most n^k. Henceforth we refer to f and the DNF computing it interchangeably. Let $m = n^a$ for some constant $a \in (0,1]$. Consider the following $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$.

1. The witness consists of indices $i_1, i_2, \ldots, i_m \in ([n^k])^m$ such that the i_jth term in the jth copy of f is satisfied.
2. The verifier, given query access to x and the witness $(i_1, i_2, \ldots, i_m)$ uses Grover's search algorithm to look for $j \in [m]$ such that the i_jth term of the jth copy of f is not satisfied.
3. If such an index is found, the verifier rejects. Else, it accepts.

If $(\mathsf{AND}_m \circ f^m)(x) = -1$ then the verifier, given the honest witness, will not find an unsatisfied term among those indicated by the witness. Thus the protocol has perfect completeness. If $(\mathsf{AND}_m \circ f^m)(x) = 1$ then there must be some j such that the jth copy of f is not satisfied.

It follows that none of the terms of f are satisfied in that copy, and so given any witness $(i_1, i_2, \ldots, i_m)$ the i_jth term of the jth copy of f will not be satisfied. By properties of Grover search, the verifier finds that index and rejects with probability $2/3$. Thus the protocol is sound. The witness length of the protocol is at most $km \log n$ since each term requires $k \log n$ bits to describe. Whether or not each term is satisfied can be computed exactly with k queries. One can thus implement the search over these checks with $O(k\sqrt{m})$ queries using standard techniques (see, e.g., [21]). This proves the upper bound.

The lower bound holds due to the proof of Theorem 15. Since $m \leq n$, $\widetilde{\deg}_{1-1/m}(f) \geq \widetilde{\deg}_{1-1/n}(f) > n^{1-\delta}$. For convenience, let $d = n^{1-\delta}$. Setting $\varepsilon = \max\{2^{-w}, 2^{-m-2}\}$, we can apply Corollary 12 as in the proof of Theorem 15. This yields a constant τ such that every (M, ε)-SBQP approximation to $\mathsf{AND}_m \circ f^m$ with $M \leq \varepsilon 2^{\tau\sqrt{m \log(1/\varepsilon)} \log m}$ must have degree $\Omega(d\sqrt{m \log(1/\varepsilon)}) = \Omega(d\sqrt{m \cdot \min\{m, w\}})$. Due to our setting of ε and m, there exists a constant $\tau'' > 0$ such that the degree lower bound holds for all (M, ε)-SBQP approximations with $M \leq 2^{\tau'' a\sqrt{m \cdot \min\{m-w\}} \log n}$. Set $\tau' = \tau'' a/2$. Suppose there is a $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length $w \leq \tau' m \log n$. Then by Lemma 14 there is a $(2^{w+2}, \varepsilon)$-SBQP approximation to $\mathsf{AND}_m \circ f^m$ with degree $O(q \min\{w, \max\{w, m\}\}) = O(qw)$. Combining the upper and lower bounds, $q = \Omega(d\sqrt{m \cdot \min\{m, w\}}/w) = \min\{\Omega(d\sqrt{m/w}), \Omega(dm/w)\}$. ◀

Our techniques also allow us to get query complexity lower bounds for batch verifying f from lower bounds on $\widetilde{\deg}_{1-1/m^c}(f)$ for arbitrarily small constants $c > 0$ and on $\widetilde{\deg}_{1/3}(f)$.

▶ **Theorem 17.** *Let n be a function of m and let $\{f : \{-1,1\}^{n(m)} \to \{-1,1\}\}_{m \in \mathbb{N}}$ be an arbitrary family of functions. The following statements hold for all sufficiently large m.*

1. *For every $c > 0$, if $\widetilde{\deg}_{1-1/m^c}(f) > d \geq 1$ then every $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length at most $O(m)$ requires $\Omega(d)$ queries.*
2. *If $\widetilde{\deg}_{1/3}(f) > d \geq 1$ then every $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length at most $m - 3$ requires $\Omega(d)$ queries.*

Proof.

(Part 1). Fix $c > 0$ and $f : \{-1,1\}^{n(m)} \to \{-1,1\}$. Assume $\widetilde{\deg}_{1-1/m^c}(f) > d \geq 1$, and suppose there is a $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length $w = O(m)$ and query complexity q. Assume that $w \geq \log(12)$; the complementary case follows immediately. Let $\varepsilon = 2^{-m-2}$. By Corollary 12 there is a constant τ that depends only on c such that every (M, ε)-SBQP approximation to $\mathsf{AND}_m \circ f^m$ with $M \leq \varepsilon 2^{\tau\sqrt{m \log(1/\varepsilon)} \log m}$ must have degree $\Omega(d\sqrt{m \log(1/\varepsilon)}) = \Omega(dm)$.

By Lemma 14 there is a degree-$O(qm)$ polynomial that $(2^{w+2}, \varepsilon)$-SBQP approximates $\mathsf{AND}_m \circ f^m$. We have $\varepsilon 2^{\tau\sqrt{m \log(1/\varepsilon)} \log m} = 2^{\Omega(m \log m)} \geq 2^{O(m)} = 2^{w+2}$. So we must have $O(qm) = \deg(p) = \Omega(dm)$, which implies $q = \Omega(d)$ as claimed.

(Part 2). Suppose $\widetilde{\deg}_{1/3}(f) > d \geq 1$, and suppose there is a $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ f^m$ with witness length $w \leq m - 3$ and query complexity q. By Corollary 13, for sufficiently large m, every $(M, 2^{-m-2})$-SBQP approximation with $M \leq 2^{m-1}$ must have degree $\Omega(dm)$. By Lemma 14 there is a degree-$O(qm)$ polynomial that $(2^{w+2}, 2^{-m-2})$-SBQP approximation to $\mathsf{AND}_m \circ f^m$. Since $w \leq m - 3$, $2^{w+2} \leq 2^{m-1}$. The query lower bound follows. ◀

4.2 $\mathrm{QMA}^{\mathrm{dt}}$ Lower Bounds for Natural Functions

Some functions have the property that they contain as sub-functions the problem of batch verifying smaller copies of themselves. For such functions, a lower bound on the $\mathrm{QMA}^{\mathrm{dt}}$ complexity of batch verification immediately implies a lower bound on the $\mathrm{QMA}^{\mathrm{dt}}$ complexity

of the functions themselves. In this subsection, we use this insight along with Theorem 17 to get $\mathrm{QMA}^{\mathrm{dt}}$ lower bounds for element distinctness, surjectivity, and k-distinctness. Our lower bounds make use of the observation by Sherstov and Thaler [41] that the surjectivity and k-element distinctness functions contain the batch verification problem for smaller instances of themselves as a subfunction.

▶ **Theorem 18.**

1. $\mathsf{QMA}^{\mathsf{dt}}(\mathsf{AND}_m \circ \mathsf{OR}_n^m) = \Omega((mn)^{1/3})$. *Consequently,* $\mathsf{QMA}^{\mathsf{dt}}(\mathsf{AND}_{n^{1/3}} \circ \mathsf{OR}_{n^{2/3}}^{n^{1/3}}) = \Omega(n^{1/3})$.
2. $\mathsf{QMA}^{\mathsf{dt}}(\mathsf{SURJ}_n) = \widetilde{\Omega}(n^{3/7})$.
3. $\mathsf{QMA}^{\mathsf{dt}}(k\mathsf{ED}_n) = \widetilde{\Omega}(n^{\frac{3k-1}{7k-1}})$.

Before presenting the proof of our lower bounds, we briefly mention lower bounds that follow directly from prior work. For $\mathsf{AND}_{n^{1/3}} \circ \mathsf{OR}_{n^{2/3}}^{n^{1/3}}$, the best known lower bound is $\Omega(n^{1/6})$, which follows – via the connection between $\mathrm{QMA}^{\mathrm{dt}}$ complexity and threshold degree in [17] – from the fact that the threshold degree of $\mathsf{AND}_{n^{1/3}} \circ \mathsf{OR}_{n^{2/3}}^{n^{1/3}}$ is $\Omega(n^{1/3})$. For SURJ_n, the best known lower bound is $\widetilde{\Omega}(n^{1/4})$, due to a threshold degree lower bound of $\widetilde{\Omega}(\sqrt{n})$ [12]. Finally, for $k\mathsf{ED}_n$, we have $\mathsf{QMA}^{\mathsf{dt}}(k\mathsf{ED}_n) \geq \mathsf{QMA}^{\mathsf{dt}}(\mathsf{ED}_n) = \widetilde{\Omega}(n^{2/5})$, which was proved implicitly in [41].

Proof of Theorem 18.

(Part 1). Note that for every m' that divides m, $\mathsf{AND}_m \circ \mathsf{OR}_n^m = \mathsf{AND}_{m'} \circ (\mathsf{AND}_{m/m'} \circ \mathsf{OR}_n^{m/m'})^{m'}$. By a lower bound from Section 2, $\mathrm{adeg}(\mathsf{AND}_{m/m'} \circ \mathsf{OR}_n^{m/m'}) = \Omega(\sqrt{mn/m'})$. By Theorem 17, every $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_{m'} \circ (\mathsf{AND}_{m/m'} \circ \mathsf{OR}_n^{m/m'})^{m'}$ with witness length at most $m' - 3$ must have query complexity $\Omega(\sqrt{mn/m'})$. Setting $m' = (mn)^{1/3}$, we get that every $\mathrm{QMA}^{\mathrm{dt}}$ protocol or $\mathsf{AND}_m \circ \mathsf{OR}_n^m$ either has witness length at least $(mn)^{1/3} - 3$ or query complexity $\Omega(\sqrt{mn/(mn)^{1/3}}) = \Omega((mn)^{1/3})$, which means $\mathsf{QMA}^{\mathsf{dt}}(\mathsf{AND}_m \circ \mathsf{OR}_n^m) = \Omega((mn)^{1/3})$.

(Part 2). Set $R = N/2$ as is typical. Observe that for all m that divide $N/2$, $\mathsf{QMA}^{\mathsf{dt}}(\mathsf{SURJ}_{N,R}) \geq \mathsf{QMA}^{\mathsf{dt}}(\mathsf{AND}_m \circ \mathsf{SURJ}_{N/m,R/m}^m)$. This is due to the fact that evaluating $\mathsf{AND}_m \circ \mathsf{SURJ}_{N/m,R/m}^m$ amounts to evaluating whether m different lists of numbers from $[R/m]$ each contain every $r \in [R/m]$. One way to do this is to, for all $i \in [m]$, add $(i-1)R/m$ to each number in the ith list, concatenate all the lists, and then evaluate $\mathsf{SURJ}_{N,R}$ on the concatenated list. It follows that any $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{SURJ}_{N,R}$ can be used to solve $\mathsf{AND}_m \circ \mathsf{SURJ}_{N/m,R/m}^m$ with no additional cost. Recall that $\mathrm{adeg}(\mathsf{SURJ}_{N/m,R/m}) = \widetilde{\Omega}((N/m)^{3/4})$. By Theorem 17, every $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ \mathsf{SURJ}_{N/m,R/m}^m$ with witness length at most $m - 3$ must have query complexity $\widetilde{\Omega}((N/m)^{3/4})$. Setting $m = N^{3/7}$, every $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ \mathsf{SURJ}_{N/m,R/m}^m$ must either have witness length at least $N^{3/7} - 3$ or query complexity at $\widetilde{\Omega}(N^{3/7})$. Hence $\mathsf{QMA}^{\mathsf{dt}}(\mathsf{SURJ}_n) = \widetilde{\Omega}(N^{3/7})$, which is $\widetilde{\Omega}(n^{3/7})$.

(Part 3). Like in the case of surjectivity, $\mathsf{QMA}^{\mathsf{dt}}(k\mathsf{ED}_{N,N}) \geq \mathsf{QMA}^{\mathsf{dt}}(\mathsf{AND}_m \circ k\mathsf{ED}_{N/m,N/m}^m)$. When one shifts the input to $\mathsf{AND}_m \circ k\mathsf{ED}_{N/m,N/m}^m$ as in part 2, the shifted input is a "yes" instance of $k\mathsf{ED}_{N,N}$ if and only if the original instance is a "yes" instance of $\mathsf{AND}_m \circ k\mathsf{ED}_{N/m,N/m}^m$. Recall that $\mathrm{adeg}(k\mathsf{ED}_{N/m,N/m}) = \widetilde{\Omega}((N/m)^{3/4-1/(4k)})$. So Theorem 17 tells us that every $\mathrm{QMA}^{\mathrm{dt}}$ protocol for $\mathsf{AND}_m \circ k\mathsf{ED}_{N/m,N/m}^m$ with witness length at most $m - 3$ must have query complexity $\widetilde{\Omega}((N/m)^{3/4-1/(4k)})$. Setting $m = n^{\frac{3k-1}{7k-1}}$ yields the desired lower bound. ◀

References

1. Scott Aaronson. Impossibility of succinct quantum proofs for collision-freeness. *Quantum Info. Comput.*, 12(1–2):21–28, 2012. doi:10.26421/QIC12.1-2-3.
2. Scott Aaronson, Robin Kothari, William Kretschmer, and Justin Thaler. Quantum lower bounds for approximate counting via Laurent polynomials. In *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, 2020.
3. Scott Aaronson and Yaoyun Shi. Quantum lower bounds for the collision and the element distinctness problems. *J. ACM*, 51(4):595–605, 2004. doi:10.1145/1008731.1008735.
4. Andris Ambainis. Polynomial degree and lower bounds in quantum complexity: Collision and element distinctness with small range. *Theory Comput.*, 1(1):37–46, 2005. doi:10.4086/TOC.2005.V001A003.
5. Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM (JACM)*, 48(4):778–797, 2001. doi:10.1145/502090.502097.
6. Nir Bitansky, Chethan Kamath, Omer Paneth, Ron D Rothblum, and Prashant Nalini Vasudevan. Batch proofs are statistically hiding. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 435–443, 2024. doi:10.1145/3618260.3649775.
7. Harry Buhrman, Richard Cleve, Ronald de Wolf, and Christof Zalka. Bounds for small-error and zero-error quantum algorithms. In *40th Annual Symposium on Foundations of Computer Science, FOCS 1999, New York, NY, USA, October 17-18, 1999*, 1999. doi:10.1109/SFFCS.1999.814607.
8. Mark Bun, Mandar Juvekar, and Samuel King. QMA lower bounds for batch verification via approximate degree. *arXiv preprint*, 2026. arXiv:2607.08888.
9. Mark Bun, Robin Kothari, and Justin Thaler. The polynomial method strikes back: Tight quantum query bounds via dual polynomials. *Theory Comput.*, 16:1–71, 2020. doi:10.4086/TOC.2020.V016A010.
10. Mark Bun and Justin Thaler. Dual lower bounds for approximate degree and Markov–Bernstein inequalities. *Inf. Comput.*, 243:2–25, 2015. doi:10.1016/J.IC.2014.12.003.
11. Mark Bun and Justin Thaler. Hardness amplification and the approximate degree of constant-depth circuits. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, 2015. doi:10.1007/978-3-662-47672-7_22.
12. Mark Bun and Justin Thaler. The large-error approximate degree of ac^0. *Theory Comput.*, 17:1–46, 2021. doi:10.4086/TOC.2021.V017A007.
13. Mark Bun and Justin Thaler. Approximate degree in classical and quantum computing. *Foundations and Trends in Theoretical Computer Science*, 15(3–4):229–423, 2022. doi:10.1561/0400000107.
14. Jeffrey Champion and David J Wu. Non-interactive zero-knowledge from non-interactive batch arguments. In *Annual International Cryptology Conference*, pages 38–71. Springer, 2023. doi:10.1007/978-3-031-38545-2_2.
15. Arka Rai Choudhuri, Abhihsek Jain, and Zhengzhong Jin. SNARGs for P from LWE. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, 2022. doi:10.1109/FOCS52979.2021.00016.
16. Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Non-interactive batch arguments for NP from standard assumptions. In *Annual International Cryptology Conference*, 2021.
17. Marcel Dall'Agnol, Tom Gur, Subhayan Moulik, and Justin Thaler. Quantum proofs of proximity. *Quantum*, 6:834, 2022. doi:10.22331/Q-2022-10-13-834.
18. Lalita Devadas, Rishab Goyal, Yael Kalai, and Vinod Vaikuntanathan. Rate-1 non-interactive arguments for batch-NP and applications. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, 2022.

19 Dmitry Gavinsky and Alexander A. Sherstov. A separation of NP and coNP in multiparty communication complexity. *Theory Comput.*, 6(1):227–245, 2010. doi:10.4086/TOC.2010.V006A010.

20 Rishab Goyal, Aditya Jain, and Shashwatha Mitra GB. Succinct arguments for batch QMA and friends under 8 rounds. In *Annual International Cryptology Conference*, 2025.

21 Peter Høyer, Michele Mosca, and Ronald de Wolf. Quantum search on bounded-error inputs. In *International Colloquium on Automata, Languages, and Programming*, 2003.

22 Yael Kalai, Alex Lombardi, Vinod Vaikuntanathan, and Daniel Wichs. Boosting batch arguments and ram delegation. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1545–1552, 2023. doi:10.1145/3564246.3585200.

23 Yael Tauman Kalai, Omer Paneth, and Lisa Yang. How to delegate computations publicly. In *Proceedings of the 51st annual ACM SIGACT symposium on theory of computing*, pages 1115–1124, 2019. doi:10.1145/3313276.3316411.

24 Yael Tauman Kalai, Vinod Vaikuntanathan, and Rachel Yun Zhang. Somewhere statistical soundness, post-quantum security, and SNARGs. In *Theory of Cryptography Conference*, 2021.

25 Inbar Kaslasi, Guy N Rothblum, Ron D Rothblum, Adam Sealfon, and Prashant Nalini Vasudevan. Batch verification for statistical zero knowledge proofs. In *Theory of Cryptography Conference*, 2020.

26 Inbar Kaslasi, Ron D Rothblum, and Prashant Nalini Vasudevan. Public-coin statistical zero-knowledge batch verification against malicious verifiers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 219–246. Springer, 2021. doi:10.1007/978-3-030-77883-5_8.

27 Or Keret, Ron D Rothblum, and Prashant Nalini Vasudevan. Doubly-efficient batch verification in statistical zero-knowledge. In *Theory of Cryptography Conference*, 2024.

28 Hartmut Klauck. On Arthur Merlin games in communication complexity. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC 2011, San Jose, California, USA, June 8-10, 2011*, 2011. doi:10.1109/CCC.2011.33.

29 Nikhil S. Mande, Justin Thaler, and Shuchen Zhu. Improved approximate degree bounds for k-distinctness. In *15th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2020, Riga, Latvia, June 9-12, 2020*, 2020. doi:10.4230/LIPIcs.TQC.2020.2.

30 Chris Marriott and John Watrous. Quantum Arthur-Merlin games. *Comput. Complex.*, 14(2):122–152, 2005. doi:10.1007/S00037-005-0194-X.

31 Changrui Mu, Shafik Nassar, Ron D Rothblum, and Prashant Nalini Vasudevan. Strong batching for non-interactive statistical zero-knowledge. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, 2024.

32 Shafik Nassar and Ronak Ramachandran. Quantum statistical witness indistinguishability. *arXiv preprint*, 2025. arXiv:2509.01945.

33 Ran Raz and Amir Shpilka. On the power of quantum proofs. In *19th Annual IEEE Conference on Computational Complexity (CCC 2004), 21-24 June 2004, Amherst, MA, USA*, 2004. doi:10.1109/CCC.2004.1313849.

34 Omer Reingold, Guy N Rothblum, and Ron D Rothblum. Constant-round interactive proofs for delegating computation. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 49–62, 2016. doi:10.1145/2897518.2897652.

35 Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Efficient batch verification for UP. In *33rd Computational Complexity Conference (CCC 2018)*, 2018. doi:10.4230/LIPIcs.CCC.2018.22.

36 Guy N Rothblum and Ron D Rothblum. Batch verification and proofs of proximity with polylog overhead. In *Theory of Cryptography Conference*, 2020.

37 Alexander A. Sherstov. The pattern matrix method. *SIAM J. Comput.*, 40(6):1969–2000, 2011. doi:10.1137/080733644.

38 Alexander A. Sherstov. Strong direct product theorems for quantum communication and query complexity. *SIAM J. Comput.*, 41(5):1122–1165, 2012. doi:10.1137/110842661.

39 Alexander A. Sherstov. Approximating the AND-OR tree. *Theory Comput.*, 9:653–663, 2013. doi:10.4086/TOC.2013.V009A020.

40 Alexander A. Sherstov. The approximate degree of DNF and CNF formulas. *SIAM J. Comput.*, 54(3):702–774, 2025. doi:10.1137/23M1557593.

41 Alexander A. Sherstov and Justin Thaler. Vanishing-error approximate degree and QMA complexity. *Chicago Journal of Theoretical Computer Science*, 2023(3), 2023.

42 Brent Waters and David J Wu. Batch arguments for NP and more from standard bilinear group assumptions. In *Annual International Cryptology Conference*, pages 433–463, 2022. doi:10.1007/978-3-031-15979-4_15.

Good Locally Testable Codes with Small Alphabet and Small Query Size

Uriya A. First[1]
University of Haifa, Israel

Stav Lazarovici
University of Haifa, Israel

Abstract

Ben-Sasson, Goldreich and Sudan [3] showed that a binary error correcting code admitting a 2-query tester cannot be good, i.e., it cannot have both linear distance and constant rate. They also showed that there are no good codes if the alphabet is a finite field $\mathbb{F}$, the code is $\mathbb{F}$-linear, and the 2-query tester is $\mathbb{F}$-linear. We show that those are essentially the only limitations on the existence of good locally testable codes (LTCs). That is, there are good 2-query LTCs on any alphabet with more than 2 letters, and good 3-query LTCs with a binary alphabet. Similarly, there are good 3-query $\mathbb{F}$-linear LTCs, and for every $\mathbb{F}$-vector space V of dimension greater than 1, there are good 2-query LTCs with alphabet V whose tester is $\mathbb{F}$-linear. This completely solves, for every $q \geq 2$ and alphabet (resp. $\mathbb{F}$-vector space) Σ, the question of whether there is a good q-query LTC (resp. $\mathbb{F}$-LTC) with alphabet Σ. Our proof builds on the recent good 2-query $\mathbb{F}$-LTCs of the first author and Kaufman [11], by establishing a general method for reducing the alphabet size of a good low-query LTC.

2012 ACM Subject Classification Mathematics of computing → Coding theory; Theory of computation → Computational complexity and cryptography

Keywords and phrases error correcting code, locally testable code, property testing

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.70

Category RANDOM

Related Version *Extended Version*: `https://arxiv.org/abs/2512.16082` [10]

Funding This research was supported by ISF grant no. 721/24.

Acknowledgements We are grateful to the anonymous referees for many useful suggestions, and to Esty Kelman for finding a mistake in an earlier version of Section 4.

1 Introduction

Throughout, Σ is a (finite) alphabet and $\mathbb{F}$ is a finite field. We let $C \subseteq \Sigma^n$ be an error correcting code, which we think of as ranging in an infinite family of codes with block length tending to infinity. The normalized Hamming distance in Σ^n is denoted $\delta(\cdot,\cdot)$. We recall other relevant coding theory terminology in Section 2.

Informally, the code $C \subseteq \Sigma^n$ is said to be a locally testable code (LTC) if it admits a randomized algorithm – called a *tester* – that can estimate with high probability whether a word $w \in \Sigma^n$ is far from C or not by reading only a constant number of letters from w. Formally, we require the tester for C to read at most q letters from w, accept all words in C, and reject every $w \in \Sigma^n - C$ with probability at least $\mu \cdot \delta(w, C)$ for some $\mu > 0$. Here, q and μ are independent of n and w. When we wish to specify the implicit constants q and μ, we will say that $C \subseteq \Sigma^n$ is a q-query LTC with soundness μ. Also, if not indicated otherwise, we assume that any tester is *non-adaptive*, i.e., it decides which positions to read from the input word before reading them.

[1] Corresponding author

© Uriya A. First and Stav Lazarovici;
licensed under Creative Commons License CC-BY 4.0

The notion of an LTC arose in the 1990s from the many works on the developing theories of *property testing* and *probabilistically checkable proofs* (PCPs). It first appeared in print in [13, Dfn. 9] in a more lax version, in which the requirement that T rejects every $w \in \Sigma^n$ with probability at least $\mu \cdot \delta(w, C)$ was imposed only for words that are $\Omega(1)$-far from C. Such a codes are known as *weak LTCs*. The definition of LTCs which we use here, known as *strong LTCs*, originates from [15, Dfn. 2.2], which intitiated the systematic study of LTCs as objects of independent interest. See [14] for a more detailed history of how LTCs emerged.

Since their introduction, it was not clear whether there are LTCs that are also *good codes*, i.e., LTCs whose relative distance and rate is bounded away from 0; see [14, Open Problem 2], for example. Indeed, Ben-Sasson, Goldreich and Sudan [3] showed that there are no good 2-query LTCs on a binary alphabet, and no $\mathbb{F}$-linear 2-query LTCs. (Here, an LTC is said to be $\mathbb{F}$-linear if its alphabet is $\mathbb{F}$ and its tester checks $\mathbb{F}$-linear constraints.) Further limitations on 2-query LTCs and 3-query LTCs appeared in [6, 18]. See also [4]. Roughly at the same time, a series of works including [23, 8, 5, 25, 19, 16] established the existence of "almost" good LTCs (both weak and strong), e.g., LTCs with rate $\mathrm{poly}(\log n)^{-1}$ and constant relative distance, or good codes admitting a tester reading $(\log n)^{O(\log \log n)}$ letters. The existence of good LTCs was finally settled by Dinur–Evra–Livne–Lubotzky–Mozes [9] and Panteleev–Kalachev [22] (independently), who showed that good $\mathbb{F}$-linear LTCs (with large query size) do exist. Shortly after, the first author and Kaufman [11] (see also [12]) showed that there are also good 2-query LTCs (with a large alphabet). The latter codes are $\mathbb{F}$-LTCs, meaning that their alphabet is an $\mathbb{F}$-vector space, and their tester checks $\mathbb{F}$-linear constraints.

Main Results

In this work, we completely resolve the question of what are the integers $q \geq 2$ and alphabets Σ for which there exists a good q-query LTC (resp. $\mathbb{F}$-LTC) with alphabet Σ.

▶ **Theorem 1.** *Let $q \geq 2$ be an integer and let Σ be an alphabet.*

(i) *There exists a good q-query LTC with alphabet Σ if and only if $(q, |\Sigma|) \neq (2, 2)$.*

(ii) *Suppose Σ is an $\mathbb{F}$-vector space. Then there exists a good q-query $\mathbb{F}$-LTC with alphabet Σ if and only if $(q, \dim \Sigma) \neq (2, 1)$.*

Note that the non-existence statements of Theorem 1 were established by Ben-Sasson, Goldreich and Sudan [3]. Theorem 1 therefore says that the restrictions on the existence of good LTCs (resp. $\mathbb{F}$-LTCs) from [3] are actually the only restrictions.

We prove the existence part of Theorem 1 by introducing an alphabet reduction method for LTCs and applying it to the good 2-query $\mathbb{F}$-LTCs of [12, Thms. 9.7, 9.8] (see also [11, Cor. 7.2] where the construction is given for $\mathbb{F} = \mathbb{F}_2$). This is the content of the following two theorems, which together with [12, Thms. 9.7, 9.8] and [3] imply Theorem 1.

▶ **Theorem 2.** *If there exists a good 2-query LTC with a possibly adaptive tester on some alphabet Σ, then there exist*

(i) *a good 2-query LTC with alphabet Δ for every Δ having more than 2 letters, and*

(ii) *a good 3-query LTC with a binary alphabet.*

▶ **Theorem 3.** *If there exists a good 2-query $\mathbb{F}$-LTC, then there exist*

(i) *a good 2-query $\mathbb{F}$-LTC with alphabet Δ for every $\mathbb{F}$-vector space Δ with $\dim \Delta \geq 2$, and*

(ii) *a good 3-query $\mathbb{F}$-linear LTC.*

In both Theorem 2 and Theorem 3, the relative distance, rate and soundness of the LTCs promised by the theorem are at least proportional to the corresponding parameters of the given LTC. In Theorem 3, all the parameters are diminished by a factor that is at most polynomial in $|\Sigma|$, while in Theorem 2, that factor is at most exponential in $|\Sigma|$; see Theorems 28 and 29. However, if the LTC given in Theorem 2 is an $\mathbb{F}_2$-LTC, then the relative distance, rate and soundness are diminished only by a factor polynomial in $|\Sigma|$; see Theorem 30. If the randomness complexity of the given LTC's tester is $r(n)$ (a function of the block length n), then the randomness complexity of the promised LTC's tester is $\max\{r(n), \log_2 n\} + O(1)$; see Remark 32.

Analogous statements hold for q-query LTCs. Again, see Theorems 28 and 29.

We hope that our alphabet reduction results will find other uses.

▶ Remark 4. Some methods for reducing the number of queries of a good LTC are also known. Specifically, by [12, Thm. 3.11, Prop. 3.6], if there exists a good q-query LTC satisfying some assumptions (which are satisfied by the good LTCs of [9]) on alphabet Σ, then there exists a good 2-query LTC with alphabet $\Sigma' \subseteq \Sigma^q$. A different construction of this kind, which also needs assumptions on the LTC, appears in [25, Cor. 3.4]. It is further known that the existence of a good *weak* $\mathbb{F}$-linear LTC implies the existence of a good weak 3-query $\mathbb{F}$-linear LTC [6, Thm. A.1]. We do not know if there is a query complexity reduction method that works for a general good LTC.

Proof Idea

Our alphabet reduction method for LTCs (Theorems 2 and 3) is simple in nature and consists of concatenating[2] the given 2-query LTC $C \subseteq \Sigma^n$ with a special code $D \subseteq \Delta^k$ (which remains fixed as n grows) and showing that the concatenated code $C \circ D \subseteq \Delta^{nk}$ is also a good 2-query LTC. While concatenation is a well-known method for reducing a good code's alphabet size, it usually fails to preserve the property of being a 2-query LTC. The novelty of our approach is in showing that the inner code $D \subseteq \Delta^k$ (and the identification of Σ with D) can be chosen so that the property of being a 2-query LTC will persist.

Let us now explain in more detail how our concatenation approach works in the context of Theorem 2(i). To begin, let us first see why $C \circ D \subseteq \Delta^{nk}$ is in general not a 2-query LTC. Let T denote the 2-query tester of C, and choose some encoding function $f : \Sigma \to D \subseteq \Delta^k$. Then each codeword of $C \circ D$ has the form

$$f(w_1)f(w_2)\cdots f(w_n) \in \Delta^{nk}$$

for some $w \in C$. Let $u \in \Delta^{nk}$ and write $u = u_1 u_2 \cdots u_n$ with $u_1, \ldots, u_n \in \Delta^k$. A natural way to test whether u is in $C \circ D$ would be to try one of the following:

(1) Check that $u_i \in D$ for some $i \in [n]$;

(2) Emulate the tester T of C: run T to get two coordinates $i, j \in [n]$ to read, find the letters $a, b \in \Sigma$ satisfying $f(a) = u_i$, $f(b) = u_j$ (or reject if there are no such a, b), and return what T would have returned upon reading a and b in positions i and j.

A naive implementation of (1) and (2) requires making k or $2k$ queries to u, so the 2-query testability is seemingly lost.

[2] Concatenation of codes is recalled in Section 3.

We solve the large number of queries in (1) by requiring that $D \subseteq \Delta^k$ is defined by 2-letter constraints. Then, instead of checking whether $u_i \in D$, we check whether u_i satisfies a random 2-letter constraint. (Note that since $D \subseteq \Delta^k$ remains fixed as n grows, we do not need the stronger assumption that $D \subseteq \Delta^k$ is included in a family of codes that is a 2-query LTC.)

Reducing the number of queries in (2) is possible under special assumptions on T and f. Given a random seed s for T, denote by $i_s, j_s \in [n]$ the coordinates queried by T, and by $T^s(w)$ the output of T on $w \in \Sigma^n$ when given the seed s. What we need from T and f is that for every s, there are coordinates $i', j' \in [k]$ (depending on s) such that one can determine $T^s(w)$ – which depends only on w_{i_s} and w_{j_s} – from the i'-th letter of $f(w_{i_s})$ and the j'-th letter of $f(w_{j_s})$. In other words, writing $f_{i'} : \Sigma \to \Delta$ for the i'-th coordinate of f (so that $f(a) = (f_1(a), \ldots, f_k(a))$), we want to be able to determine $T^s(w)$ from $f_{i'}(w_{i_s})$ and $f_{j'}(w_{j_s})$ for some i', j' depending on s. When this holds for every seed s, we say that T is *f-compatible*; see Section 4 for details. Provided that T is f-compatible, we can emulate the action of T^s on $u = u_1 \cdots u_n$ by reading just two letters from u, namely, $(u_{i_s})_{i'}$ and $(u_{j_s})_{j'}$, and thus perform (2) using only 2 queries.

We show in Theorem 8 that if $D \subseteq \Delta^k$ is defined by 2-letter constraints and T is f-compatible, then the concatenated code $C \circ D$ is indeed a 2-query LTC. With this at hand, we need to find a code $D \subseteq \Delta^k$ and an encoding function $f : \Sigma \to D$ such that (i) T is f-compatible, and (ii) $D \subseteq \Sigma^k$ is defined by 2-letter constraints.

The code $D \subseteq \Delta^k$ that we choose for the task is a generalization of the *long code* of [2, §3]. Formally, letting $f_1, \ldots, f_k$ denote *all* the functions from Σ to Δ (so that $k = |\Delta|^{|\Sigma|}$), we choose D to be the code $L(\Sigma, \Delta) := \{(f_1(a), \ldots, f_k(a)) \mid a \in \Sigma\} \subseteq \Delta^k$ and take $f : \Sigma \to D$ to be the obvious encoding $f(a) = (f_1(a), \ldots, f_k(a))$. We call $L(\Sigma, \Delta)$ a *generalized long code*; the usual long code is the special case where $\Sigma = \{0,1\}^t$ and $\Delta = \{0,1\}$. The rationale behind this choice of D is that having any function $g : \Sigma \to \Delta$ in the collection $\{f_i\}_{i=1,\ldots,k}$ helps in securing that T is f-compatible. (This is still not enough to guarantee f-compatibility, and we solve this below.) The drawback of choosing D to be $L(\Sigma, \Delta)$ is that it *a priori* it does not fulfill the requirement of being defined by 2-letter constraints. We prove this nontrivial statement in Theorem 17 under the assumption $|\Delta| \geq 3$ (the hardest case is $|\Delta| = 3$). By contrast, the "usual" long code $L(\{0,1\}^t, \{0,1\})$ is *not* defined by 2-letter constraints; see Remark 20.

In order to finish, it remains to check that the tester T is f-compatible. We show in Proposition 23 that the f-compatibility of T is equivalent to another condition called *Δ-separability*. While this condition fails in general, we show in Theorem 26 that T can be replaced with a Δ-separable 2-query tester with soundness that is proportional to that of T. Conveniently, this replacement is also the usual trick to turn an adaptive tester into a nonadaptive tester, so we can also treat non-adaptive 2-query testers. After that replacement, we can finally conclude that the concatenated code $C \circ D \subseteq \Delta^{nk}$ is a 2-query LTC.

The proof of Theorem 3(i) follows a similar path except that we define $D \subseteq \Delta^k$ by choosing $f_1, \ldots, f_k$ to be all the $\mathbb{F}$-linear functions from Σ to Δ. This gives rise to what we call a *generalized Hadamard code*; see § 5.2. Similarly to the generalized long code, it can be defined by 2-letter constraints if and only if $\dim \Delta \geq 2$ (Theorem 16).

▶ Remark 5. Passing from $C \subseteq \Sigma^n$ to $C \circ D \subseteq \Delta^{nk}$ as described above scales down the soundness by a constant factor depending on Σ and D. There are three sources for this scale-down.

The first is the soundness of the 2-query tester for D (cf. (1) above) – we use the naive lower bound $\frac{1}{R}$ where R is the number of 2-letter constraints defining D; for the generalized long code this factor is $|\Delta|^{-O(|\Sigma|)}$. This can probably be improved significantly. That is,

when $|\Delta| \geq 3$, we expect that the generalized long code $L(\Sigma, \Delta) \subseteq \Delta^{|\Delta|^{|\Sigma|}}$ should have a 2-query tester with soundness that is *independent of* $|\Sigma|$. Similarly, in the linear case, we expect the generalized Hadamard code (§ 5.2) to have the same property when $\dim \Delta \geq 2$. We pose this as a problem.

The second source for loosing on the soundness is the block length k of $D \subseteq \Delta^k$. This $\frac{1}{k}$ factor comes up in the analysis of the tester of $C \circ D$, and is forced in situations in which the coordinates $i', j' \in [k]$ from the definition of f-compatibility (which are determined by the random seed of T) are very unevenly distributed. See Remark 9 for details and possible ways one might overcome it (which were not pursued in this work).

The final cause for loosing on the soundness is the replacement of T by a Δ-separable tester. In more detail, in the non-linear case, the trick is to first guess what are the two letters that will be read by T, and apply T only if these letters are actually encountered (otherwise accept the word). This multiplies the soundness by a factor of $|\Sigma|^{-2}$. We do not know how to avoid this factor in general. However, in the linear case, T is morally close to being Δ-separable (it is always Δ^m-separable for $m = \lceil \frac{2 \dim \Sigma}{\dim \Delta} \rceil$), and one can use a different trick which reduces the soundness by a factor of $O(\log |\Sigma|)$.

Relation to Other Works on Alphabet Reduction

Concatenation with the "ordinary" long code and Hadamard code was used in the literature for alphabet reduction in the construction of PCPs and 2-query LTCs. For example, in [8, §§7–8], resp. [20, §6.4], this technique is used to construct non-linear, resp. linear, weak 2-query LTCs having linear distance and inverse-polylogarithmic rate. The same method was pushed further in [25] to construct (strong) $\mathbb{F}_2$-linear 3-query LTCs (with a binary alphabet) and $\mathbb{F}_2$-LTCs with an 8-letter alphabet having similar distance and rate.

In a nutshell, the idea of these alphabet reductions is to encode both the letters and the constraints of the code; this is different from our approach which encodes only the letters. To explain this, let us describe an oversimplified version of the alphabet reduction of [8]: We start with a code $C \subseteq \Sigma^n$ having a 2-query tester T. Choose some embedding of Σ in $\{0,1\}^t$ and use it to view words in Σ^n as strings of tn bits. Think of T as checking at random one of (say) R 2-letter constraints which define $C \subseteq \Sigma^n$. Since $\Sigma \subseteq \{0,1\}^t$, every such constraint is a circuit on $2t$ bits. Now construct a new code $C' \subseteq \{0,1\}^N$ by modifying the codewords $w \in C$ as follows. First, view w as a tn-bit string using that $\Sigma \subseteq \{0,1\}^t$. Then, for each of the R constraints defining C, attach to w the long-code encoding of the $2t$ bits participating in that constraint.[3] This results in a string of $N = tn + 2^{2^{2t}} R$ bits, and we take C' to be the collection of all such strings. Since the long-code encoding of $2t$ bits includes the evaluation of any circuit on those bits, if $w' \in \{0,1\}^N$ is obtained from some $w \in \Sigma^n$ by the above procedure, then we can emulate the action of T on w by reading just 1 bit from w'. The long code[4] has two additional relevant properties. First, it has a 3-query tester. Second, it has a 2-local decodability property – if $y \in \{0,1\}^{2^{2^{2t}}}$ is close to the encoding of some $x \in \{0,1\}^{2t}$, then there is a randomized algorithm which can determine x_i with high probability by reading just 2 letters from y.[5] Using these properties and other mild assumptions, it can be shown that $C' \subseteq \{0,1\}^N$ has a 3-query tester with soundness

[3] This is a little different from [8, §7], where a certain puncturing of the long code is used.
[4] More precisely, its puncturing that is used in [8, §7].
[5] The algorithm: choose uniformly at random two functions $f, g : \{0,1\}^{2t} \to \{0,1\}$ such that $f + g$ is the i-th coordinate function $e_i : \{0,1\}^{2t} \to \{0,1\}$, and return $y_f + y_g$. See [8, Thm. 7.1] for a precise statement. See [20, Lem. 6.21] for a corresponding statement for the Hadamard code.

that is proportional to that of T. Briefly, the idea is that given $u \in \{0,1\}^N$, we can detect if u is very corrupted in its last $2^{2^{2t}}R$ bits (the long-code encoding of the constraints) using the 3-query tester of the long code. If that is not the case, then we can use the 2-local decodability property of the long code to check if the last $2^{2^{2t}}R$ bits of u are consistent with the first tn bits (this requires querying $2+1=3$ bits from u). When they are almost consistent, meaning that u is very close to a word coming from some $w \in \Sigma^n$, we can emulate T on w.

There are two reasons why we cannot use this alphabet reduction approach to prove our main results. First, the above construction increases the block length by $O(R)$, where $\log_2 R$ is the randomness complexity of T, while leaving the message length the same. Therefore, applying it to a good LTC would result in a good LTC only if $R = O(n)$, equiv. the randomness complexity of T is $\log n + O(1)$. By contrast, our alphabet reduction method (Theorems 2 and 3) makes no assumptions on the randomness complexity. Second, the alphabet reduction just described (loosely) relies on the fact that the long code (resp. Hadamard code) is 2-locally decodable, which is one reason why the resulting tester needs to query $2+1=3$ letters (the other reason is that the long code's tester queries 3 letters). If we were to use this approach to get a 2-query LTC, we would need to replace the long code with a 1-locally decodable code, and there are no such useful codes. That said, it seems likely that applying the alphabet reduction of [8], [20], [25] to a good 2-query LTC (or even any good LTC) whose tester has randomness complexity $\log_2(n) + O(1)$, e.g., those of [11] (or [9], [22]), would result in a good 3-query LTC on a binary alphabet.

Finally, in terms of soundness, the alphabet reduction of [8], etc. is more economical than ours as it decreases the soundness by a constant factor (independent of $|\Sigma|$); see Remark 5 for what causes the soundness loss in our alphabet reduction. As for rate and distance, our alphabet reduction reduces the rate by a smaller factor than that of *op. cit.*, and both alphabet reductions reduce the distance by a constant factor independent of $|\Sigma|$.

Organization

The paper is organized as follows: Section 2 is preliminary and recalls necessary definitions and facts about codes and testers. In Section 3, we recall concatenation of codes. In Section 4, we prove a theorem giving sufficient conditions for the concatentation of two codes to be a q-query LTC. The inner codes to which this result will be applied – the generalized long code and the generalized Hadamard code – are presented in Section 5. In that section, it is also shown that these codes are defined by 2-letter constraints. Section 6 is concerned with showing that one can replace the tester of any LTC by another tester to which our result about concatenation of LTCs can be applied. Finally, in Section 7, all previous results are combined to prove Theorems 2 and 3.

2 Preliminaries

2.1 General Conventions

Throughout this paper, $\mathbb{F}$ is a finite field, and $[n]$ denotes the set $\{1,\dots,n\}$. Vector spaces are over $\mathbb{F}$ and are assumed to be finite-dimensional. (Unlike some other texts, the letter q will be reserved for the number of queries performed by a tester and will *not* denote the size of $\mathbb{F}$.) An alphabet is a finite set with at least two elements.

Recall that a distribution on a countable set X is a collection of non-negative real numbers $p = (p_x)_{x \in X}$ adding up to 1; the number p_x is the probability of drawing x when sampling an element from X according to p. We write $x \sim p$ to indicate that $x \in X$ is chosen at random accroding to the distribution p. For a finite set X, we write $x \sim X$ to denote that x is chosen from X uniformly at random.

2.2 Error Correcting Codes

Let Σ be an alphabet and $n \in \mathbb{N}$. We write Σ^n for the set of n-letter words in the alphabet Σ, and unless indicated otherwise, write w_i for the i-th letter of a word $w \in \Sigma^n$. As usual, the normalized Hamming distance on Σ^n, denoted $\delta(\cdot,\cdot)$, is given by $\delta(u,v) = \frac{1}{n} \cdot \#\{i \in \{1,\dots,n\} : u_i \neq v_i\}$ for all $u, v \in \Sigma^n$.

In this work, an *error correcting code*, or a *code* for short, with alphabet Σ and block length n is a nonempty subset $C \subseteq \Sigma^n$. Recall that the *relative distance* of $C \subseteq \Sigma^n$ is

$$\delta(C) := \min\{\delta(u,v) \,|\, u, v \in C,\, u \neq v\}$$

and its *rate* is

$$r(C) := \frac{\log_{|\Sigma|} |C|}{n}.$$

We say that C has relative distance δ (resp. rate r) when $\delta(C) \geq \delta$ (resp. $r(C) \geq r$) and add the word "exactly" to indicate that equality holds.

It is common to think of a code $C \subseteq \Sigma^n$ as ranging in a *family of codes*, i.e., a sequence of codes $\{C_i \subseteq \Sigma^{n_i}\}_{i \in \mathbb{N}}$ with n_i tending to ∞. Recall that the family $\{C_i \subseteq \Sigma^{n_i}\}_{i \in \mathbb{N}}$ is said to be a *good code* if there are $r, \delta > 0$ such that $r(C_i) \geq r$ and $\delta(C_i) \geq \delta$ for all i. In this case, we also say that the family $\{C_i \subseteq \Sigma^{n_i}\}_{i \in \mathbb{N}}$ has relative distance δ and rate r.

2.3 Testers

Let $q \in \mathbb{N}$. Recall that a *q-query tester*, or a *q-tester* for short, for a code $C \subseteq \Sigma^n$ is a randomized algorithm T which, given oracle access to some $w \in \Sigma^n$, reads at most q letters from w and returns 1 – meaning "accept" – or 0 – meaning "reject" – subject to the requirement that any $w \in C$ is accepted. The tester T is said to have soundness μ ($\mu \geq 0$) if

$$\Pr(T(w) = 0) \geq \mu\, \delta(w, C) \qquad \forall w \in \Sigma^n.$$

The tester T is called *non-adaptive* if it does not use information from previous queries to w to determine which position to read next.

Unless explicitly indicated, we always assume that testers are non-adaptive.

Observe that in order to specify a (non-adaptive) q-tester for $C \subseteq \Sigma^n$, it is enough to give the following data: a finite set I, a probability distribution $p = (p_i)_{i \in I}$ on I, a q-tuple $(a_1^{(i)}, \dots, a_q^{(i)}) \in [n]^q$ for each $i \in I$, and a function $T^{(i)} : \Sigma^q \to \{0,1\}$ for every $i \in I$. The corresponding tester T then works as follows: Given $w \in \Sigma^n$, choose $i \in I$ according to the distribution p, read the letters in positions $a_1^{(i)}, \dots, a_q^{(i)}$ from w and return $T^{(i)}(w_{a_1^{(i)}}, \dots, w_{a_q^{(i)}})$. We write this simply as

$$T = (T^{(i)}, (a_1^{(i)}, \dots, a_q^{(i)}), p_i)_{i \in I}.$$

A code $C \subseteq \Sigma^n$ is said to be *defined by q-letter constraints* if it has a q-query tester with positive soundness. This is equivalent to saying that there is a list of constraints on words in Σ^n, each involving q or less letters, such that C is the set of words in Σ^n satisfying all those constraints.

A family of codes $\{C_i \subseteq \Sigma^{n_i}\}_{i\in\mathbb{N}}$ is called a *q-query locally testable code* (q-query LTC) if there is $\mu > 0$ such that each code in the family admits a q-tester with soundness μ. In this case, we also say that the q-query LTC $\{C_i \subseteq \Sigma^{n_i}\}_{i\in\mathbb{N}}$ has soundness μ. A *locally testable code* (LTC) is a family of codes that is a q-query LTC for some $q \in \mathbb{N}$.

2.4 Linear Codes

Let $\mathbb{F}$ be a finite field, and suppose that the alphabet Σ is also an $\mathbb{F}$-vector space (hence $\Sigma \neq 0$). A code $C \subseteq \Sigma^n$ is said to be an *$\mathbb{F}$-code* if C is a subspace of Σ^n. An *$\mathbb{F}$-linear code* is an $\mathbb{F}$-code with alphabet $\mathbb{F}$.

Let C be an $\mathbb{F}$-code. We say that a q-tester $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i\in I}$ is *$\mathbb{F}$-linear* if for each $i \in I$, there is a linear subspace $V_i \subseteq \Sigma^q$ such that

$$T^{(i)}(u) = \begin{cases} 1 & u \in V_i \\ 0 & u \notin V_i \end{cases}$$

for all $u \in \Sigma^q$. For example, when $\Sigma = \mathbb{F}$ and V_i has codimension 1 in Σ^q, the test $T^{(i)}$ merely checks that the input $w \in \mathbb{F}^n$ satisfies a single homogeneous linear equation involving at most q of the coordinates.

Any q-tester for an $\mathbb{F}$-code can be replaced by an $\mathbb{F}$-linear q-query tester having soundness greater than or equal to the soundness of the original tester.

A *q-query $\mathbb{F}$-LTC* (resp. *$\mathbb{F}$-linear LTC*) is a family of $\mathbb{F}$-codes (resp. $\mathbb{F}$-linear codes) for which there is $\mu > 0$ such that each code in the family admits an $\mathbb{F}$-linear q-tester with soundness μ.

3 Concatenation of Codes

We proceed by recalling concatenation of codes, setting notation along the way. Let $C \subseteq \Sigma^n$ be a code with alphabet Σ, let Δ be another alphabet, and let $f : \Sigma \to \Delta^k$ be an injective function with image D. We think of D as a code inside Δ^k and of $f : \Sigma \to D \subseteq \Delta^k$ as its encoding function. Recall that the *concatenated code* $C \circ D \subseteq \Delta^{kn}$ is obtained by encoding the letters of each codeword in C using the code D. Formally,

$$C \circ D := \{f(w_1) \cdots f(w_n) \,|\, w \in C\} \subseteq \Delta^{kn}.$$

The codes C and D are often called the *outer code* and *inner code*, respectively. When the encoding map f is not clear from the context, we shall write $C \circ_f D$ for $C \circ D$. For more details, see [24, Chp. 12] or [17, §10.1], for instance.

Note that if C is an $\mathbb{F}$-code, Δ is an $\mathbb{F}$-vector space and $f : \Sigma \to \Delta^k$ is $\mathbb{F}$-linear, then $C \circ D$ is an $\mathbb{F}$-code. When $\Delta = \mathbb{F}$, the code $C \circ D$ is moreover $\mathbb{F}$-linear.

The behavior of the relative distance and rate of $C \circ D$ is well-known and summarized in the following proposition.

▶ **Proposition 6.** *With notation as above, we have*

$$\delta(C \circ D) \geq \delta(C)\delta(D) \qquad \text{and} \qquad r(C \circ D) = r(C)r(D).$$

In applications of concatenation in this paper, the outer code C will range over a family of codes $\{C_i \subseteq \Sigma^{n_i}\}_{i\in\mathbb{N}}$, while the inner code $D \subseteq \Delta^k$ will remain fixed. Then, by Proposition 6, the family $\{C_i \circ_f D \subseteq \Sigma^{kn_i}\}_{i\in\mathbb{N}}$ will be a good code as long as $\{C_i \subseteq \Sigma^{n_i}\}_{i\in\mathbb{N}}$ is a good code; this holds no matter how poor the relative distance and rate of D are.

4 Concatenation of Codes with Testers

In general, the concatenation of codes admitting q-testers with positive soundness does not share the same property. However, in this section, we will show that under certain assumptions, this is indeed the case.

▶ **Definition 7.** *Let $C \subseteq \Sigma^n$ be a code, let $T = (T^{(i)}, (a_1^{(i)}, \dots, a_q^{(i)}), p_i)_{i \in I}$ be a q-tester for C (see § 2.3), and let $f : \Sigma \to \Delta^k$ be an injective function. We denote the j-coordinate of f as $f_j : \Sigma \to \Delta$. We say that T is* f-compatible *if for every $i \in I$, there are $b_1, \dots, b_q \in [k]$ and a function $g : \Delta^q \to \{0,1\}$ such that for every $w_1, \dots, w_q \in \Sigma$,*

$$T^{(i)}(w_1, \dots, w_q) = g(f_{b_1}(w_1), \dots, f_{b_q}(w_q)).$$

In the other words, we can compute $T^{(i)}(w_1, \dots, w_q)$ by reading q letters from the word $f(w_1) \cdots f(w_q)$ in Δ^{qk}.

▶ **Theorem 8.** *Let $C \subseteq \Sigma^n$ be a code admitting a q_C-tester T_C with soundness $\mu_C > 0$. Let $f : \Sigma \to \Delta^k$ be an injective function such that the code $D := \mathrm{im}(f) \subseteq \Delta^k$ has a q_D-tester T_D with soundness $\mu_D > 0$. Suppose further that T_C is f-compatible. Then the concatenated code $E := C \circ_f D \subseteq \Delta^{nk}$ admits a $\max\{q_C, q_D\}$-tester T_E with soundness*

$$\frac{\mu_C \mu_D}{qk + \mu_C + \mu_D}.$$

Proof. We abbreviate q_C to q and write $T = T_C = (T^{(i)}, (a_1^{(i)}, \dots, a_q^{(i)}), p_i)_{i \in I}$. Given $i \in I$, denote the $b_1, \dots, b_q \in [k]$ and the function $g : \Delta^q \to \{0,1\}$ promised by Definition 7 by $b_1^{(i)}, \dots, b_q^{(i)} \in [k]$ and $g^{(i)} : \Delta^q \to \{0,1\}$.

Fix some distribution $\rho = (\rho_1, \rho_2, \rho_3)$ on the set $\{1,2,3\}$; we will specify ρ at the end. Using ρ, we define a tester T_E for $E \subseteq \Delta^{nk}$ as follows: Given $u \in \Delta^{nk}$, write $u = u_1 \cdots u_n$ with $u_1, \dots, u_n \in \Delta^k$. Then choose $x \in \{1,2,3\}$ according to the distribution ρ and perform routine (x) from the following list.

(1) Choose $\ell \in [n]$ uniformly at random and return $T_D(u_\ell)$ (q_D letters are read from u).

(2) Choose $i \in I$ according to $p = (p_i)_{i \in I}$ and return $g^{(i)}\left((u_{a_1^{(i)}})_{b_1^{(i)}}, \dots, (u_{a_q^{(i)}})_{b_q^{(i)}}\right)$ (q_C letters are read from u).

(3) Choose $i \in I$ according to $p = (p_i)_{i \in I}$, choose $\ell \in [q_C]$ uniformly at random and return $T_D(u_{a_\ell^{(i)}})$ (q_D letters are read from u).

The rationale behind (1) and (2) was explained in the introduction. Routine (3) is included in order to deal with situations where $a_1^{(i)}, \dots, a_q^{(i)}$ are very unevenly distributed; otherwise it is subsumed by routine (1).

It is straightforward to see that T_E accepts every word in E. It remains to prove that it has the claimed soundness.

Let $u \in \Delta^{nk}$ and write $\varepsilon := \delta(u, E)$. Then

$$\Pr(T(u) = 0) = \sum_{x=1}^{3} \rho_x \cdot \Pr(T(u) = 0 \,|\, (x) \text{ is executed}).$$

We will bound the three summands on the right hand side from below. To that end, we think of D^n as a subset of Δ^{nk} and define the following words in Δ^{nk}:

- u' is an element of D^n with minimal distance from u.
- u'' is an element of E with minimal distance from u'.

We further let $w' \in \Sigma^n$ denote the word satisfying $f^n(w') = u'$. Obseve that

$$\varepsilon = \delta(u, E) \le \delta(u, u'') \le \delta(u, u') + \delta(u', u''). \tag{4.1}$$

Now, since T_D has soundness μ_D, we have

$$\begin{aligned}\Pr(T_E(u) = 0 \,|\, (1) \text{ is exec.}) &= \Pr_{\ell\sim[n]}\left(T_D(u_\ell) = 0\right) \ge \frac{1}{n}\sum_{\ell=1}^{n} \mu_D\, \delta_{\Delta^k}(u_\ell, D) \\ &= \mu_D\, \delta_{\Delta^{nk}}(u, D^n) = \mu_D\, \delta(u, u').\end{aligned}$$

Next, denote by R the probability that $u_{a_t^{(i)}} \in D$ for all $t = 1, \dots, q$ (equivalently, $u_{a_t^{(i)}} = u'_{a_t^{(i)}}$ for $t = 1, \dots, q$) when $i \in I$ is chosen according to the distribution p. Then

$$\begin{aligned}\Pr(T_E(u) = 0 \,|\, (2) \text{ is exec.}) &= \Pr_{i\sim p}\left[g^{(i)}\left((u_{a_1^{(i)}})_{b_1^{(i)}}, \dots, (u_{a_q^{(i)}})_{b_q^{(i)}}\right) = 0\right] \\ &\ge \Pr_{i\sim p}\left[g^{(i)}\left((u_{a_1^{(i)}})_{b_1^{(i)}}, \dots, (u_{a_q^{(i)}})_{b_q^{(i)}}\right) = 0 \text{ and } u_{a_t^{(i)}} = u'_{a_t^{(i)}}\ \forall t \in [q]\right] \\ &= \Pr_{i\sim p}\left[g^{(i)}\left((u'_{a_1^{(i)}})_{b_1^{(i)}}, \dots, (u'_{a_q^{(i)}})_{b_q^{(i)}}\right) = 0 \text{ and } u_{a_t^{(i)}} = u'_{a_t^{(i)}}\ \forall t \in [q]\right] \\ &\ge \Pr_{i\sim p}\left[T^{(i)}(w'_{a_1^{(i)}}, \dots, w'_{a_q^{(i)}}) = 0\right] - \Pr_{i\sim p}\left[\exists t \in [q] \,:\, u_{a_t^{(i)}} \ne u'_{a_t^{(i)}}\right] \\ &\ge \mu_C\, \delta(w', C) - (1 - R) \\ &\ge \mu_C\, \delta(u', E) - (1 - R) = \mu_C\, \delta(u', u'') - (1 - R),\end{aligned}$$

while

$$\begin{aligned}\Pr(T_E(u) = 0 \,|\, (3) \text{ is exec.}) &= \Pr_{i\sim p, \ell\sim[q]}(T_D(u_{a_\ell^{(i)}}) = 0) \\ &\ge (1 - R) \cdot \Pr_{i\sim p, \ell\sim[q]}\left[T_D(u_{a_\ell^{(i)}}) = 0 \,\middle|\, u_{a_\ell^{(i)}} \notin D \text{ for some } \ell\right] \\ &\ge (1 - R) \cdot \frac{1}{q} \cdot \frac{\mu_D}{k}\end{aligned}$$

Here, the second-to-last inequality holds since $u_{a_\ell^{(i)}} \notin D$ implies $\delta_{\Delta^k}(u_{a_\ell^{(i)}}, D) \ge \frac{1}{k}$.

Putting everything together gives

$$\begin{aligned}\Pr(T_E(u) = 0) &\ge \rho_1 \cdot \delta(u, u')\mu_D + \rho_2 \cdot (\mu_C\, \delta(u', u'') - (1 - R)) + \rho_3 \cdot (1 - R)\frac{\mu_D}{qk} \\ &= \rho_1\mu_D\delta(u, u') + \rho_2\mu_C\, \delta(u', u'') + (1 - R)\left(-\rho_2 + \rho_3 \cdot \frac{\mu_D}{qk}\right) =: (\star)\end{aligned}$$

Suppose that ρ is chosen so that $\rho_2 \le \frac{\mu_D}{qk}\rho_3$. Then by (4.1), we have

$$(\star) \ge \rho_1\mu_D\delta(u, u') + \rho_2\mu_C\, \delta(u', u'') \ge \varepsilon \cdot \min\{\rho_1\mu_D, \rho_2\mu_C\}.$$

Subject to the requirements $\rho_1 + \rho_2 + \rho_3 = 1$ and $\rho_2 \le \frac{\mu_D}{qk}\rho_3$, the right hand side is maximized for

$$\rho_1 = \frac{\mu_C}{\mu_C + \mu_D + qk}, \quad \rho_2 = \frac{\mu_D}{\mu_C + \mu_D + qk}, \quad \rho_3 = \frac{qk}{\mu_C + \mu_D + qk},$$

in which case it evaluates to $\frac{\mu_C\mu_D}{\mu_C + \mu_D + qk} \cdot \varepsilon$. This is what we want. ◀

▶ Remark 9. Without additional assumptions in Theorem 8, the tester T_E described in its proof cannot have soundness larger than $\Omega(\frac{1}{k})$. The reason is that we make no assumption

on the distribution of the coordinates $b_1^{(i)}, \ldots, b_q^{(i)} \in [k]$ from Definition 7 as $i \in I$ distributes according to p. For example, consider an extreme situation in which $b_1^{(i)} = \cdots = b_q^{(i)} = 1$ for all $i \in I$ and $\delta(D) > \frac{1}{k}$. Start with a word $w' \in \Sigma^n$ that is δ-far from C, let $w \in C$ be the closest word to w', and let $u = f(w_i) \cdots f(w_n)$ and $u'_i = f(w'_i) \cdots f(w'_n)$. Then, for every $i \in [n]$ such that $w_i \neq w'_i$, replace the $(ki+1)$-letter of u' with the corresponding letter in u; denote the resulting word by u''. Now, the word u'' satisfies $\delta(u'', C \circ D) \geq \delta(u', C \circ D) - \delta(u'', u') \geq \delta \cdot (\delta(D) - \frac{1}{k}) = \Omega(\delta)$. However, when we attempt to emulate T_C on u'' (routine (2) above), we cannot distinguish between u and u''. Thus, the only way we could detect that u'' is not in $C \circ D$ is by applying the tester of D to one of the k-letter blocks of u'' (routines (1) or (3)). Regardless of how this block is chosen, it differs by only one letter from a word in D, and so we are only guaranteed to succeed with probability $\mu_D \cdot \frac{1}{k}$ or less. It follows that T_E rejects u'' with probability $O(\frac{1}{k})$, while u'' is $\Omega(\delta)$-far from $C \circ D$.

One can overcome the $\Omega(\frac{1}{k})$ limitation by requiring that for every $t \in [q]$, the position $k(a_t^{(i)} - 1) + b_t^{(i)}$ distributes nearly uniformly in $[nk]$ when $i \sim p$. This requires imposing additional assumptions on T, f, D. We omit the details because these assumptions are not guaranteed in the situations in which we apply Theorem 8.

Another approach to bypass this limitation is to take into advantage the fact that in routine (3) above, we only need to check that the block $u_{a_t^{(i)}}$ was not corrupted *in position* $b_t^{(i)}$. Calling $T_D(u_{a_t^{(i)}})$ is wasteful since it does not use this information. To use it, however, we need D to be equipped with a "local tester" able to detect with high probability whether a given word is far from D *or corrupted in a given position*. Such local testers were not considered in the literature, and are out of the scope of this work.

Theorem 8 implies in particular that if $C \subseteq \Sigma^n$ admits a q-tester with soundness $\mu > 0$ and if Δ is an alphabet containing Σ, then the code $C \subseteq \Delta^n$ (i.e., C thought of as a code in Δ^n) has a q-tester with soundness $\frac{\mu}{q+1+\mu}$ (take $f : \Sigma \to \Delta^1$ to be the embedding of Σ in Δ). In this case, however, there is a tester with larger soundness.

▶ **Proposition 10.** *Let $C \subseteq \Sigma^n$ be a code admitting a q-tester T with soundness $\mu > 0$, and let Δ be an alphabet containing Σ. Then $C \subseteq \Delta^n$ admits a tester T' with soundness $\frac{\mu}{\mu+1}$.*

Proof. Write $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i \in I}$. Fix $\rho_1, \rho_2 > 0$ with $\rho_1 + \rho_2 = 1$ and consider the following tester T' for $C \subseteq \Delta^n$: Given $u \in \Delta^n$, choose $x \in \{1, 2\}$ according to the distribution (ρ_1, ρ_2) and perform routine (x) from the following:

(1) Choose $\ell \in [n]$ uniformly at random and accept u if and only if $u_\ell \in \Sigma$.

(2) Choose $i \in I$ according to p. Read $u_{a_1^{(i)}}, \ldots, u_{a_q^{(i)}}$. If one of these letters is not in Σ, then reject u. Otherwise, return $T^{(i)}(u_{a_1^{(i)}}, \ldots, u_{a_q^{(i)}})$.

As in the proof of Theorem 8, to show that T' has the claimed soundness, let $u' \in \Sigma^n$ be a word with minimal distance from u, and let $u'' \in C$ be a word with minimal distance from u'. Then

$$\delta(u, u') + \delta(u', u'') \geq \delta(u, C) =: \varepsilon.$$

Moreover, $\Pr(T'(u) = 0 \,|\, (1) \text{ exec.}) = \delta(u, u')$. Next, as in the proof of Theorem 8, writing R for the probability that $u_{a_1^{(i)}}, \ldots, u_{a_q^{(i)}} \in \Sigma$ when $i \in I$ distributes according to p, we have

$$\begin{aligned}\Pr(T'(u) = 0 \,|\, (2) \text{ is exec.}) &= (1-R) \\ &\quad + \Pr_{i\sim p}\left[T^{(i)}(u_{a_1^{(i)}}, \ldots, u_{a_q^{(i)}}) = 0 \text{ and } u_{a_1^{(i)}}, \ldots, u_{a_q^{(i)}} \in \Sigma\right] \\ &\geq (1-R) + [\mu\,\delta(u', C) - (1-R)] \geq \mu\,\delta(u', u'').\end{aligned}$$

It follows that

$$\Pr(T'(u) = 0) \geq \rho_1\,\delta(u, u') + \rho_2\mu\,\delta(u', u'') \geq \min\{\rho_1, \rho_2\mu\}\varepsilon.$$

Taking $\rho_1 = \frac{\mu}{\mu+1}$ and $\rho_2 = \frac{1}{\mu+1}$ completes the proof of proposition. ◀

▶ Remark 11. In Theorem 8, if Σ and Δ are $\mathbb{F}$-vector spaces, $f : \Sigma \to \Delta$ is $\mathbb{F}$-linear, and the code C is an $\mathbb{F}$-code, then $E = C \circ D \subseteq \Delta^{nk}$ is also an $\mathbb{F}$-code. Moreover, it is clear from the proof that if the testers T_C and T_D are $\mathbb{F}$-linear, then so is the tester T_E of E.

Similarly, in Proposition 10, if Σ and Δ are $\mathbb{F}$-vector spaces such that Σ is a subspace of Δ and T is $\mathbb{F}$-linear, then so is the tester T' for $C \subseteq \Delta^n$.

5 The Generalized Hadamard Code and the Generalized Long Code

In this section, we introduce two types of codes generalizing the (linear) Hadamard code and the (non-linear) long code, respectively. They will ultimately serve as the inner code when we apply Theorem 8, and to that end, we will show that they can be defined using 2-letter constraints. This stands in contrast to the original Hadamard code and long code which, as we also show, cannot be defined by 2-letter constraints.

5.1 A General Construction

We begin with a general construction which is natural in the context of Theorem 8. Let S be a finite set, let Δ be an alphabet and let $f_1, \ldots, f_n : S \to \Delta$ be some functions. We associate with S, Δ and the collection $\{f_i\}_{i=1}^n$ the code

$$D(\{f_i\}_{i=1}^n) := \{(f_1(s), \ldots, f_n(s)) \,|\, s \in S\} \subseteq \Delta^n.$$

If we further define $f : S \to \Delta^n$ by $f(s) = (f_1(s), \ldots, f_n(s))$ and assume that f is injective, then this recovers the setting of Theorem 8. In fact, every code $D \subseteq \Delta^n$ equipped with an encoding function $f : S \to D$ is of the form $D(\{f_i\}_{i=1}^n)$, but the point of our construction is to build a code in Δ^n from functions $f_1, \ldots, f_n : S \to \Delta$ rather than doing the opposite.

The construction recovers two well-known codes. When $S = \{0,1\}^k$, $\Delta = \{0,1\}$ and $f_1, \ldots, f_n$ consist of all the functions from S to $\{0,1\}$ (so that $n = 2^{2^k}$), the code C is known as the *long code*, e.g., see [2, §3]. Next, when $S = V$ with V an $\mathbb{F}$-vector space, $\Delta = \mathbb{F}$ and $f_1, \ldots, f_n$ are all the $\mathbb{F}$-linear functions from V to $\mathbb{F}$ (so that $n = |V|$), the code C is (up to equivalence) the $|\mathbb{F}|$-ary *Hadamard code* associated to the vector space V, e.g., see [17, Exercise 2.22] (we also recall the definition in § 5.2).

We now introduce a q-tester for $D(\{f_i\}_{i=1}^n)$. To that end, we say that a collection of functions $f_1, \ldots, f_q : S \to \Delta$ is *dependent* if

$$\mathrm{im}(f_1 \times \cdots \times f_q) = \{(f_1(s), \ldots, f_q(s)) \,|\, s \in S\} \subsetneq \Delta^q.$$

Informally, this means that knowing (say) the values of $f_1, \ldots, f_{q-1}$ on some input $s \in S$ gives some information on the value of f_q on that input, even without knowing what s is. (Note that it is possible for a single function f_1 to be dependent, namely, if its image is strictly smaller than Δ.) With this at hand, we introduce:

▶ **Definition 12** (q-dependence tester)**.** *With notation as a above, given a word $w \in \Delta^n$, the* q-dependence tester *for $D(\{f_i\}_{i=1}^n)$ chooses uniformly at random a tuple $(i_1, \ldots, i_q) \in [n]^q$ such that $f_{i_1}, \ldots, f_{i_q}$ are dependent, and accepts w if and only if $(w_{i_1}, \ldots, w_{i_q}) \in \mathrm{im}(f_{i_1} \times \cdots \times f_{i_q})$.*

If the q-dependence tester has positive soundness, then this soundness is at least n^{-q}.

▶ **Example 13.** To demonstrate how the q-dependence tester works, suppose for simplicity that $\Delta = \mathbb{F}$, and let $w \in \Delta^k$ be a word that is always accepted by the 3-dependence tester. We claim that if f_i, f_j, f_k satisfy $f_i + f_j = f_k$, then we must have $w_i + w_j = w_k$. Indeed, this is forced by the fact that $\mathrm{im}(f_i \times f_j \times f_k) \subseteq \{(\alpha, \beta, \alpha+\beta) \,|\, \alpha, \beta \in \mathbb{F}\}$. Similarly, if $f_i = \alpha f_j$ for some $\alpha \in \mathbb{F}$, then $w_i = \alpha w_j$, because for any $k \in [n]$, $\mathrm{im}(f_i, f_j, f_k) \subseteq \{(\beta, \alpha\beta, \gamma) \,|\, \beta, \gamma \in \mathbb{F}\}$ (f_k was added artificially to f_i and f_j to make a dependent triple of functions). More generally, for a general Δ, if there is a function $g : \Delta \times \Delta \to \Delta$ such that $g \circ (f_i, f_j) = f_k$ (i.e. $g(f_i(s), f_j(s)) = f_k(s)$ for all $s \in S$), then $g(w_i, w_j) = w_k$.

The following useful property of the q-dependence tester will be used freely in the sequel.

▶ **Proposition 14.** *With notation as above, the code $D(\{f_i\}_{i=1}^n) \subseteq \Delta^n$ is defined by q-letter constraints (i.e., it has a q-tester with positive soundness) if and only if its q-dependence tester has positive soundness.*

Proof. The "if" direction is clear, so we treat the "only if" direction. Write $D := D(\{f_i\}_{i=1}^n)$ and let $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i \in I}$ be a q-tester for D having positive soundness. Let $w \in \Delta^n - D$. It is enough to prove that w fails the q-dependence test with some positive probability. Since T has positive soundness, there is $i \in I$ such that $T^{(i)}(w_{a_1^{(i)}}, \ldots, w_{a_q^{(i)}}) = 0$. Since T accepts all words in D, we must also have $T^{(i)}(f_{a_1^{(i)}}(s), \ldots, f_{a_q^{(i)}}(s)) = 1$ for all $s \in S$. Consequently, $(w_{a_1^{(i)}}, \ldots, w_{a_q^{(i)}}) \notin \mathrm{im}(f_{a_1^{(i)}} \times \cdots \times f_{a_q^{(i)}})$. This means that $f_{a_1^{(i)}}, \ldots, f_{a_q^{(i)}}$ are dependent and that w will be rejected if the q-dependence test chooses these functions. ◀

5.2 The Generalized Hadamard Code

Keeping the notation above, suppose that both S and Δ are $\mathbb{F}$-vector spaces. We write V in place of S to denote that. Let $f_1, \ldots, f_n$ enumerate all the $\mathbb{F}$-linear functions from V to Δ (so that $n = |\mathbb{F}|^{\dim V \cdot \dim \Delta} = |\Delta|^{\dim V} = |V|^{\dim \Delta}$). The *generalized Hadamard code* associated to V and Δ is defined to be

$$H(V, \Delta) := D(\{f_i\}_{i=1}^n) = \{(f_1(v), \ldots, f_n(v)) \,|\, v \in V\} \subseteq \Delta^n.$$

It easy to see that $H(V, \Delta) \subseteq \Delta^n$ is an $\mathbb{F}$-code of relative distance $1 - \frac{1}{|\Delta|}$ and rate $\frac{\log_{|\Delta|} |V|}{|\Delta|^{\dim V}} = \frac{\dim V}{\dim \Delta \cdot |\Delta|^{\dim V}} = \frac{\log_{|\Delta|} n}{n \dim \Delta}$.

▶ Remark 15. Recall that the Hadamard code associated to an $\mathbb{F}$-vector space V is $\mathrm{Hom}_{\mathbb{F}}(V, \mathbb{F})$ – the set of linear functions from V to $\mathbb{F}$ – viewed as a subset of $\mathbb{F}^V$, i.e., all functions from V to $\mathbb{F}$. More concretely, writing $v_1, \ldots, v_n$ for the vectors in V, the Hadamard code of V is

$$H' = \{(f(v_1), \ldots, f(v_n)) \,|\, f : V \to \mathbb{F} \text{ is linear}\} \subseteq \mathbb{F}^n.$$

This code is in fact equivalent to $H(V, \mathbb{F})$, which is the reason why we call $H(V, \Delta)$ a generalized Hadamard code. Indeed, let V^* denote the dual vector space of V. Then $f_1, \ldots, f_n$ are the vectors of V^*. Since every linear map g from V^* to $\mathbb{F}$ admits a unique $v \in V$ such that $g(f) = f(v)$ for all $f \in V^*$, the code H' is just $H(V^*, \mathbb{F})$. Choosing an isomorphism $V \to V^*$ then gives an equivalence between H' and H.

It is well-known that the code $H(V,\mathbb{F})$ is 3-testable. Indeed, it is equivalent to the ordinary Hadamard code $\mathrm{Hom}_{\mathbb{F}}(V,\mathbb{F}) \subseteq \mathbb{F}^V$, which is defined by 3-letter constraints, namely,

$$\mathrm{Hom}_{\mathbb{F}}(V,\mathbb{F}) = \{f : V \to \mathbb{F} \,:\, f(x+y) = f(x) + f(y) \text{ and } f(\alpha x) = \alpha f(x) \text{ for all } x, y \in V,\ \alpha \in \mathbb{F}\}.$$

Tracing this back to $H(V,\mathbb{F}) \subseteq \mathbb{F}^n$ ($n = |V|$) shows that a word $w \in \mathbb{F}^n$ is in $H(V,\mathbb{F})$ if and only if

(1) $w_i = w_j + w_k$ for all $i,j,k \in [n]$ such that $f_i = f_j + f_k$ and

(2) $w_i = \alpha w_j$ for all $i,j \in [n]$ and $\alpha \in \mathbb{F}$ such that $f_i = \alpha f_j$.

Consequently, the 3-dependence tester of $H(V,\mathbb{F})$ has positive soundness (cf. Example 13). (In fact, for some fields $\mathbb{F}$, e.g. $\mathbb{F}_2$, the Hadamard code is known to have 3-testers with soundness that is independent of V; see [7], [1] and the many followup works.) On the other hand, it is not difficult to see that the 2-dependence tester for $H(V,\mathbb{F})$ has soundness 0 if $\dim V > 1$; the reason is that a pair of $\mathbb{F}$-linear functions $f, g : V \to \mathbb{F}$ is dependent if and only if they are proportional. As a result, $H(V,\mathbb{F})$ is not defined by 2-letter constraints (Proposition 14). By contrast, we now show that $H(V,\Delta)$ is defined by 2-letter constraints when $\dim \Delta \geq 2$.

▶ **Theorem 16.** *Let Δ be a vector space of dimension > 1 and let V be any vector space. Then the 2-dependence tester (Definition 12) for the code $H(V,\Delta) \subseteq \Delta^n$ has positive soundness (depending on n).*

Proof. We may assume without loss of generality that $\Delta = \mathbb{F}^m$ for some $m > 1$. Given functions $g_1, \ldots, g_r \in \mathrm{Hom}_{\mathbb{F}}(V,\mathbb{F})$ ($r \leq m$), we denote by $(g_1, \ldots, g_r, \vec{0})$ the function from V to $\Delta = \mathbb{F}^m$ sending $v \in V$ to $(g_1(v), \ldots, g_r(v), 0, \ldots, 0) \in \mathbb{F}^m = \Delta$. Let $g_1, \ldots, g_t$ enumerate all the linear functions from V to $\mathbb{F}$ ($t = |V|$). We may assume that $f_1, \ldots, f_n$ are numbered so that $f_i = (g_i, \vec{0})$ whenever $1 \leq i \leq t$.

Let $w \in \Delta^n$ be a word that is always accepted by the 2-dependence tester of $H(V,\Delta)$. We need to show that $w \in H(V,\Delta)$.

Since $\mathrm{im}(f_1), \ldots, \mathrm{im}(f_t) \subseteq \mathbb{F} \times \{0\}^{m-1}$ and w passes the 2-dependence test, we must have $w_1, \ldots, w_t \in \mathbb{F} \times \{0\}^{m-1}$. Thus, for each $i \in [t]$, we can write $w_i = (u_i, 0, \ldots, 0)$ with $u_i \in \mathbb{F}$.

Let $u = u_1 \cdots u_t$. We claim that $u \in D(\{g_i\}_{i=1}^t) = H(V,\mathbb{F})$. As we noted earlier, in order to show this, it is enough to check that for all $i,j,k \in [t]$ with $g_i + g_j = g_k$ (equiv. $f_i + f_j = f_k$), we have $u_i + u_j = u_k$, and for all $i,j \in [t]$ and $\alpha \in \mathbb{F}$ with $g_i = \alpha g_j$ (equiv. $f_i = \alpha f_j$), we have $u_i = \alpha u_j$. The second statement holds because w passes the 2-dependence test, so we only need to establish the first. There is $\ell \in [n]$ such that $f_\ell = (g_i, g_j, \vec{0})$. Then $\mathrm{im}(f_i \times f_\ell) \subseteq \{((\alpha,0,\ldots,0),(\alpha,\beta,0,\ldots,0)) \,|\, \alpha,\beta \in \mathbb{F}\}$, and so our assumption on w implies that w_ℓ has the form $(u_i, *, 0, \ldots, 0)$. Repeating this argument with f_j in place of f_i gives $w_\ell = (u_i, u_j, 0, \ldots, 0)$. Now, since $\mathrm{im}(f_k \times f_\ell) \subseteq \{((\alpha+\beta,0,\ldots,0),(\alpha,\beta,0,\ldots,0)) \,|\, \alpha,\beta \in \mathbb{F}\}$, we must have $u_k = u_i + u_j$, which is what we want. We conclude that $u \in H(V,\mathbb{F})$.

Let $v \in V$ be a vector such that $u_i = g_i(v)$ for all $i \in [t]$. We finish the proof by showing that $w_i = f_i(v)$ for all $i \in [n]$. Let $i \in [n]$. Then there are $i_1, \ldots, i_m \in [t]$ such that $f_i = (g_{i_1}, \ldots, g_{i_m})$. Now, for every $\ell \in [m]$, the dependence between f_i and f_{i_ℓ} implies that the ℓ-th component of $w_i \in \mathbb{F}^m$ is $u_{i_\ell} = g_{i_\ell}(v)$. As this holds for every ℓ, it follows that $w_i = (g_{i_1}(v), \ldots, g_{i_m}(v)) = f_i(v)$, as required. ◀

5.3 The Generalized Long Code

Returning to the general setting of § 5.1, let $f_1, \ldots, f_n$ denote all functions from S to Δ (so that $n = |\Delta|^{|S|}$). We call

$$L(S, \Delta) := D(\{f_i\}_{i=1}^n) = \{(f_1(s), \ldots, f_n(s)) \,|\, s \in S\} \subseteq \Delta^n$$

the *generalized long code* associated to S and Δ. This code has relative distance $1 - \frac{1}{|\Delta|}$ and rate $\frac{\log_{|\Delta|} |S|}{|\Delta|^{|S|}} = \frac{\log_{|\Delta|} \log_{|\Delta|} n}{n}$. As we noted earlier, this is a generalization of the *long code*, which arises as the special case where $S = \{0,1\}^k$ and $\Delta = \{0,1\}$.

We will show in Remark 20 that $L(S, \Delta)$ cannot be defined by 2-letter constraints when $|\Delta| = 2$. However, it turns out that it can be defined by 2-letter constraints when $|\Delta| > 2$.

▶ **Theorem 17.** *Let S be a set and let Δ be an alphabet with more than 2 letters. Then the 2-dependence tester of the generalized long code $L(S, \Delta) \subseteq \Delta^n$ has positive soundness (depending on n).*

In order to prove the theorem, we first note that $L(S, \{0,1\})$ has a 3-tester with positive soundness.

▶ **Proposition 18.** *With notation as above, suppose that $\Delta = \mathbb{F}_2$ and the functions $f_1, \ldots, f_n$ from S to Δ are numbered so that f_1 is the function sending every element of S to 1. Then a word $w \in \mathbb{F}_2^n$ is in $L(S, \Delta)$ if and only if*
(1) *$w_i + w_j = w_k$ (in $\mathbb{F}_2$) for every $i, j, k \in [n]$ such that $f_i + f_j = f_k$,*
(2) *$w_i \cdot w_j = w_k$ for every $i, j, k \in [n]$ such that $f_i \cdot f_j = f_k$, and*
(3) *$w_1 = 1$.*
In particular, $L(S, \mathbb{F}_2)$ is defined by 3-letter constraints.

Proof. This is well-known for $S = \{0,1\}^k$ [2, Prp. 3.2]. For the general case, see the extended version of this paper [10, Prp. 5.7]. ◀

We shall also need the following key lemma.

▶ **Lemma 19.** *Let $g_1, \ldots, g_t : S \to \{0,1\}$ be distinct functions. For every $i, j \in [t]$, let $g_{i,j}$ and $g'_{i,j}$ denote the functions from S to $\{0,1,2\}$ defined as follows:*

$$g_{i,j}(s) = \begin{cases} 0 & f_i(s) = 0 \\ 1 & f_i(s) = 1 \wedge f_j(s) = 0 \\ 2 & f_i(s) = 1 \wedge f_j(s) = 1 \end{cases} \qquad g'_{i,j}(s) = \begin{cases} 1 & f_i(s) = 1 \\ 0 & f_i(s) = 0 \wedge f_j(s) = 1 \\ 2 & f_i(s) = 0 \wedge f_j(s) = 0 \end{cases}$$

Let $f_1, \ldots, f_n : S \to \{0,1,2\}$ enumerate the functions in $\{g_i, g_{i,j}, g'_{i,j} \,|\, i, j \in [t]\}$. If the code $D' := D(\{g_i\}_{i=1}^t) \subseteq \{0,1\}^t$ is defined by 3-letter constraints, then $D := D(\{f_j\}_{j=1}^n) \subseteq \{0,1,2\}^n$ is defined by 2-letter constraints.

Proof. We may assume that $f_1, \ldots, f_n$ are numbered so that $f_i = g_i$ when $i \in [t]$.

Suppose $w \in \{0,1,2\}^n$ always passes the 2-dependence test of D. We need to show that $w \in D$. We begin by noting that since $\mathrm{im}(f_i) \subseteq \{0,1\}$ for all $i \in [t]$, we must have $w_i \in \{0,1\}$ for $i \in [t]$.

Next, fix $i, j \in [t]$ and let $k, \ell \in [n]$ be the numbers for which $f_k = g_{i,j}$ and $f_\ell = g'_{i,j}$. Observe that $f_i = g_i$ and $f_k = g_{i,j}$ are dependent, and $f_j = g_j$ and $f_k = g_{i,j}$ are dependent. These dependencies and our assumption on w imply that (i) $w_k = 0$ if and only if $w_i = 0$, (ii) if $w_k = 1$, then $w_j = 0$, and (iii) if $w_k = 2$, then $w_j = 1$. This means that

$$w_k = \begin{cases} 0 & w_i = 0 \\ 1 & w_i = 1 \wedge w_j = 0 \\ 2 & w_i = 1 \wedge w_j = 1. \end{cases} \tag{5.1}$$

Arguing similarly with ℓ in place of k gives

$$w_\ell = \begin{cases} 1 & w_i = 1 \\ 0 & w_i = 0 \wedge w_j = 1 \\ 2 & w_i = 0 \wedge w_j = 0. \end{cases} \tag{5.2}$$

At this point, if we could show that $w' := w_1 \cdots w_t$ is in D', then we could conclude that $w \in D$. Indeed, if it were the case that $w' \in D'$, then there would be an $s \in S$ such that $w_i = g_i(s) = f_i(s)$ for all $i \in [t]$. Since for every $p \in [n] - [t]$, the function f_p is either $g_{i,j}$ or $g'_{i,j}$ for some $i, j \in [t]$, by the previous paragraph, we would have that w_p is $g_{i,j}(s)$ or $g'_{i,j}(s)$, respectively, and consequently $w_p = f_p(s)$.

Suppose for the sake of contradiction that $w' \notin D'$. By assumption, D' admits a 3-tester $T = (T^{(u)}, (a_u, b_u, c_u), p_u)_{u \in U}$ with positive soundness. Then there is some $u \in U$ such that $T^{(u)}(w_{a_u}, w_{b_u}, w_{c_u}) = 0$. Write $(i, j, p) = (a_u, b_u, c_u)$ and $(x_1, x_2, x_3) = (w_{a_u}, w_{b_u}, w_{c_u})$. By the pigeonhole principle, at least two of x_1, x_2, x_3 are equal. We reorder (a_u, b_u, c_u) to have $x_1 = x_2$. Suppose that $x_1 = x_2 = 1$. The fact that $T^{(u)}(v_i, v_j, v_p) = 1$ for all $v \in D'$ while $T^{(u)}(x_1, x_2, x_3) = 0$ means that

$$\mathrm{im}(f_i \times f_j \times f_p) \subseteq \{0,1\}^3 - \{(x_1, x_2, x_3)\} = \{0,1\}^3 - \{(1,1,x_3)\}.$$

As a result, $\mathrm{im}(g_{i,j} \times f_p) \subseteq \{0,1,2\} \times \{0,1\} - \{(2, x_3)\}$. Writing $g_{i,j} = f_k$ with $k \in [n]$, this means that f_k and f_p are dependent (as functions from S to $\{0,1,2\}$), and thus $(w_k, w_p) \in \{0,1,2\} \times \{0,1\} - \{(2, x_3)\}$. However, since $w_i = x_1 = 1$ and $w_j = x_2 = 1$, we have $w_k = 2$ by (5.1), and since $w_p = x_3$, it follows that $(w_k, w_p) = (2, x_3)$, a contradiction to our earlier conclusion. Similarly, when $x_1 = x_2 = 0$, we reach a contradiction using (5.2). As all possibilities lead to contradiction, our assumption $w' \notin D'$ must have been false. This completes the proof. ◀

We are now ready to prove Theorem 17.

Proof of Theorem 17. We may assume without loss of generality that $\{0,1,2\} \subseteq \Delta$ and that the functions $f_1, \ldots, f_n : S \to \Delta$ are numbered so that $f_1, \ldots, f_t$ ($t = 2^{|S|}$) are all the functions from S to $\{0,1\}$.

Suppose that $w \in \Delta^n$ always passes the 2-dependence test. We need to show that $w \in L(S, \Delta)$.

We begin by showing that the exists $s \in S$ such that $w_i = f_i(s)$ for all $i \in [t]$. To that end, let $g_i = f_i$ for $i \in [t]$, and for all $i, j \in [t]$, define $g_{i,j}, g'_{i,j} : S \to \{0,1,2\}$ as in Lemma 19. We may assume that there is $m \in \{t, \ldots, n\}$ such that $f_1, \ldots, f_m$ enumerate all the functions in the set $\{g_i, g_{i,j}, g'_{i,j} \mid i, j \in [t]\}$. Now, since w passes the 2-dependence, and since $\mathrm{im}(f_i) \subseteq \{0,1,2\}$ for all $i \in [m]$, we have $w_i \in \{0,1,2\}$ for all $i \in [m]$. We may therefore think of $w' := w_1 \cdots w_m$ as a word in $\{0,1,2\}^m$. Our assumption on w implies that w' also passes the 2-dependence test for the code $D(\{f_i\}_{i=1}^m)$, so by Lemma 19 and Proposition 18 (applied to $D(\{f_i\}_{i=1}^t) = L(S, \{0,1\})$), we have $w' \in D(\{f_i\}_{i=1}^m)$. This means in particular that there is $s \in S$ such that $w_i = f_i(s)$ for all $i \in [t]$ (and even for all $i \in [m]$).

We finish by showing that $w_i = f_i(s)$ for all $i \in [n]$. Fix some $i \in [n]$. For every $a \in \Delta$, let $g_a : \Delta \to \{0,1\}$ be the function mapping a to 1 and all other elements of Δ to 0. There is some $j \in [t]$ such that $f_j = g_a \circ f_i$. The dependency between f_i and f_j and our assumption on w now imply that $w_j = g_a(w_i)$ (cf. Example 13). Since $w_j = f_j(s)$, we get that $g_a(w_i) = f_j(s) = g_a(f_i(s))$. By the definition of g_a, this means that $w_i = a$ if and only if $f_i(s) = a$. As this holds for every $a \in \Delta$, we conclude that $w_i = f_i(s)$. ◀

▶ Remark 20. By contrast to Theorem 17, the code $L(S, \{0,1\}) \subseteq \{0,1\}^n$ ($n = 2^{|S|}$) is not defined by 2-letter constaints when $|S| > 2$. To show this, it is enough to exhibit a word $w \in \{0,1\}^n$ that always passes the 2-dependence test and is not in $L(S, \{0,1\})$. Fix distinct $x_1, x_2, x_3 \in S$ and define the word $w \in \{0,1\}^n$ by $w_i = \mathrm{maj}(f_i(x_1), f_i(x_2), f_i(x_3))$ for all $i \in [n]$, where $\mathrm{maj}(-)$ is the majority function. The word w is not in $L(S, \{0,1\})$ because for every $s \in S$, there is $i \in [n]$ such that $f_i(s) \neq \mathrm{maj}(f_i(x_1), f_i(x_2), f_i(x_3)) = w_i$.

Let us now show that w always passes the 2-dependence test. Suppose that $f_i, f_j : S \to \{0,1\}$ are dependent. We need to show that $(w_i, w_j) \neq (a, b)$ for every $(a, b) \in \{0,1\}^2 - \mathrm{im}(f_i \times f_j)$. Fix such (a, b). If $w_i \neq a$, then it is clear that $(w_i, w_j) \neq (a, b)$. On the other hand, if $w_i = a$, then $\mathrm{maj}(f_i(x_1), f_i(x_2), f_i(x_3)) = a$, and so at least 2 of $f_i(x_1), f_i(x_2), f_i(x_3)$ equal a. Since $(a, b) \notin \mathrm{im}(f_i \times f_j)$, at least 2 of $f_j(x_1), f_j(x_2), f_j(x_3)$ are different from b. This means that $w_j \neq b$ and again we get $(w_i, w_j) \neq (a, b)$, as required.

When $|S| = 2$, the long code $L(S, \{0,1\})$ is $\{0101, 0011\} \subseteq \{0,1\}^4$ (up to equivalence), and it is easy to see that it is defined by 2-letter constaints.

6 Separable Testers

In order to apply Theorem 8 and change the alphabet of an LTC from Σ to Δ, we need to have a suitable function $f : \Sigma \to \Delta^k$ such that the tester T of our code is f-compatible (Definition 7). In this section, we give a necessary and sufficient condition on T to be compatible with *some* f, and moreover show that every tester can be replaced with one that satisfies our condition and has proportional soundness.

▶ **Definition 21.** *Let $C \subseteq \Sigma^n$ be a code and let $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i \in I}$ be a q-tester for C. Let Δ be another alphabet. We say that the tester T is* Δ-separable *if for every $i \in I$, there are functions $g_1, \ldots, g_q : \Sigma \to \Delta$ such that $T^{(i)} : \Sigma^q \to \{0,1\}$ factors via the function $g_1 \times \cdots \times g_q : \Sigma^q \to \Delta^q$, i.e., there is $g : \Delta^q \to \{0,1\}$ such that $T^{(i)} = g \circ (g_1 \times \cdots \times g_q)$.*

When C is an $\mathbb{F}$-code, T is $\mathbb{F}$-linear, and Δ is an $\mathbb{F}$-vector space, we say that T is linearly Δ-separable *if $g_1, \ldots, g_q$ can be taken to be $\mathbb{F}$-linear maps.*

▶ **Example 22.** Suppose that $C \subseteq \Sigma^n$ is an $\mathbb{F}$-code and $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i \in I}$ is q-tester such that each $T^{(i)}$ checks a single linear constraint involving at most q letters. Then T is linearly $\mathbb{F}$-separable. Indeed, by assumption, for every $i \in I$, there is a linear map $\varphi_i : \Sigma^q \to \mathbb{F}$ such that for every $w \in \Sigma^q$, we have $T^{(i)}(w) = 1$ if and only if $\varphi_i(w) = 0$. For such φ_i, there are linear functionals $g_1, \ldots, g_q : \Sigma \to \mathbb{F}$ such that $\varphi_i(w) = g_1(w_1) + \cdots + g_q(w_q)$. In particular, we can determine from $g_1(w_1), \ldots, g_q(w_q)$ the value of $T^{(i)}(w)$, meaning that $T^{(i)}$ factors via $g_1 \times \cdots \times g_q : \Sigma^q \to \mathbb{F}^q$.

We now show that Δ-separability is a necessary and sufficient condition for a tester T to be f-compatible with some $f : \Sigma \to \Delta^k$.

▶ **Proposition 23.** *Let $C \subseteq \Sigma^n$ be a code, let T be a q-tester for C, and let Δ be another alphabet.*

- **(i)** *If there is a function $f : \Sigma \to \Delta^k$ such that T is f-compatible, then T is Δ-separable.*
- **(ii)** *If T is Δ-separable, then there is $k \in \mathbb{N}$ and an injective function $f : \Sigma \to \Delta^k$ such that T is f-compatible. In fact, f can be constructed as follows: Let $k = |\Delta^\Sigma|$, let $f_1, \ldots, f_k$ be all the functions from Σ to Δ, and take $f(w) = (f_1(w), \ldots, f_k(w))$.*

Proof. Write $T = (T^{(i)}, (a_1^{(i)}, \dots, a_q^{(i)}), p_i)_{i \in I}$.

(i) For $j \in [k]$, let $f_j : \Sigma \to \Delta$ denote the j-th component of f. Let $i \in I$. Our assumption that T is f-compatible means that there exist a function $g : \Delta^q \to \{0,1\}$ and $b_1, \dots, b_q \in [k]$ such that for every $w_1, \dots, w_q \in \Sigma$, we have $T^{(i)}(w_1, \dots, w_q) = g(f_{b_1}(w_1), \dots, f_{b_q}(w_q))$. This is equivalent to saying that $T^{(i)} = g \circ (f_{b_1} \times \cdots \times f_{b_q})$, so $T^{(i)}$ factors via $f_{b_1} \times \cdots \times f_{b_q} : \Sigma^q \to \Delta^q$, and we have shown that T is Δ-separable.

(ii) Define f as in the statement. We need to show that T is f-compatible. Let $i \in I$. Then there are functions $g_1, \dots, g_q : \Sigma \to \Delta$ and a function $g : \Delta^q \to \{0,1\}$ such that $g \circ (g_1 \times \cdots \times g_q) = T^{(i)}$. By the construction of f, there are $b_1, \dots, b_q \in [k]$ such that $g_\ell = f_{b_\ell}$ for all $\ell \in [q]$. Then $g \circ (f_{b_1} \times \cdots \times f_{b_q}) = T^{(i)}$, or rather, $T^{(i)}(w_1, \dots, w_q) = g(f_{b_1}(w_1), \dots, f_{b_q}(w_q))$ for all $w \in \Sigma^q$. This is exactly what we need to show. ◀

▶ **Proposition 24.** *Let $C \subseteq \Sigma^n$ be an $\mathbb{F}$-code, let T be an $\mathbb{F}$-linear q-tester for T, and let Δ be a nonzero $\mathbb{F}$-vector space.*

(i) *If there is an $\mathbb{F}$-linear function $f : \Sigma \to \Delta^k$ such that T is f-compatible, then T is linearly Δ-separable.*

(ii) *If T is linearly Δ-separable, then there is $k \in \mathbb{N}$ and an injective function $f : \Sigma \to \Delta^k$ such that T is f-compatible. In fact, f can be constructed as follows: Let $k = |\operatorname{Hom}_{\mathbb{F}}(\Sigma, \Delta)|$, let $f_1, \dots, f_k$ be all the linear functions from Σ to Δ, and take $f(w) = (f_1(w), \dots, f_k(w))$.*

Proof. This is completely analogous to the proof of Proposition 23. ◀

▶ Remark 25. In Proposition 23(ii), the image of $f : \Sigma \to \Delta^k$ is the generalized long code $L(\Sigma, \Delta)$ of § 5.3. In Proposition 24(ii), the image of $f : \Sigma \to \Delta^k$ is the generalized Hadamard code $H(\Sigma, \Delta)$ of § 5.2.

Next, we show that every code admitting a q-tester also admits a Δ-separable q-tester with soundness that is proportional to that of the original tester. In particular, when given an LTC, we can always assume that its tester is Δ-separable.

▶ **Theorem 26.** *Let $C \subseteq \Sigma^n$ be a code, let T be a possibly adaptive q-tester for C with soundness $\mu > 0$, and let Δ be another alphabet. Then C admits a Δ-separable tester T' with soundness $\frac{\mu}{|\Sigma|^q}$.*

This proof is a mild modification of a standard trick to turn T into a non-adaptive q-tester. The idea is to guess what will be the q letters that the adaptive tester T would see. Based on this guess, we can determine which coordinates T would read, and then read letters in those positions. If the letters did match our guess, we return what T would return, and if our guess was wrong, we accept the word.

Proof. Let I be the set of random seeds for T (say, $I = \{0,1\}^r$ if T flips r coins). We define a non-adaptive tester T' as follows: Given $w \in \Sigma^n$, choose $u = (u_1, \dots, u_q) \in \Sigma^q$ and $i \in I$ uniformly at random. Let $a_1^{(i,u)}, \dots, a_q^{(i,u)}$ be the coordinates that T would have read on the seed i if the letters it would query in these positions were $u_1, u_2, \dots$, and let c be the output of T in that situation. Now query $w_{a_1^{(i,u)}}, \dots, w_{a_q^{(i,u)}}$. If this sequence is different from $u_1, \dots, u_q$, then w is accepted. Otherwise, T' returns c.

It is straightforward to see that $\Pr(T'(w)=0) \geq \frac{1}{|\Sigma^q|}\Pr(T(w)=0)$, so T' has soundness $\frac{\mu}{|\Sigma|^q}$. It remains to show that T' is Δ-separable. To that end, note that

$$T' = \left(T^{(i,u)}, (a_1^{(i,u)}, \ldots, a_q^{(i,u)}), \tfrac{1}{|I|\cdot|\Sigma|^q}\right)_{(i,u)\in I\times\Sigma^q},$$

where $T^{(i,u)} : \Sigma^q \to \{0,1\}$ a function that returns 1 for every $u' \in \Sigma^q - \{u\}$ and for u returns what T would have returned on the random seed i if it would read the letters $u_1, u_2, \ldots$ in positions $a_1^{(i,u)}, a_2^{(i,u)}, \ldots$.

Fix $(i,u) \in I \times \Sigma^q$. We need to show that $T^{(i,u)} = g \circ (g_1 \times \cdots \times g_q)$ for some $g_1, \ldots, g_q : \Sigma \to \Delta$ and $g : \Delta^q \to \{0,1\}$. Without loss of generality, we may assume that $0, 1 \in \Delta$. If $T^{(i,u)}(u) = 1$, then $T^{(i,u)}$ is identically 1 and we can take $g_1, \ldots, g_q, g$ to be the functions which map everything to 1. Otherwise, $T^{(i,u)}(u) = 0$ and $T^{(i,u)}(w) = 1$ for all $w \in \Sigma^q - \{u\}$. In this case, define $g_\ell : \Sigma \to \Delta$ by

$$g_\ell(w) = \begin{cases} 0 & w_\ell = u_\ell \\ 1 & w_\ell \neq u_\ell \end{cases}$$

and let $g : \Delta^q \to \{0,1\}$ be the function which maps $(0,\ldots,0)$ to 0 and all other elements to 1. It is straightforward to check that $T^{(i,u)} = g \circ (g_1 \times \cdots \times g_q)$, so T' is indeed Δ-separable. ◀

▶ **Theorem 27.** *Let $C \subseteq \Sigma^n$ be an $\mathbb{F}$-code, let T be an $\mathbb{F}$-linear q-tester for C with soundness $\mu > 0$, and let Δ be a nonzero $\mathbb{F}$-vector space. Then C admits a linearly Δ-separable tester T' with soundness $\left\lceil \frac{q \dim \Sigma}{\dim \Delta} \right\rceil^{-1} \mu$.*

Proof. Write $T = (T^{(i)}, (a_1^{(i)}, \ldots, a_q^{(i)}), p_i)_{i\in I}$ and put $m = \lceil \frac{q\dim\Sigma}{\dim\Delta} \rceil$. Then for every $i \in I$, there is a subspace $V_i \subseteq \Sigma^q$ such that $T^{(i)}(u) = 1$ if and only if $u \in V_i$ ($u \in \Sigma^q$). By the definition of m, we have $\dim \Delta^m = m \dim \Delta \geq \dim \Sigma^q$. Thus, there is a linear map $h_i : \Sigma^q \to \Delta^m$ with kernel V_i. For $j \in [m]$, denote by $h_{ij} : \Sigma^q \to \Delta$ the j-th component of h_i.

We now define an $\mathbb{F}$-linear q-tester T' as follows: Given $w \in \Sigma^n$, choose $i \in I$ according to p and $j \in [m]$ uniformly at random, and accept w if and only if $h_{ij}(w_{a_1^{(i)}}, \ldots, w_{a_q^{(i)}}) = 0$. It is clear that T' accepts every $w \in C$. Moreover, since $V_i = \ker h_i = \bigcap_{j=1}^m \ker h_{ij}$, whenever $T^{(i)}(w_{a_1^{(i)}}, \ldots, w_{a_q^{(i)}}) = 0$, there is at least one $j \in [m]$ such that $h_{ij}(w_{a_1^{(i)}}, \ldots, w_{a_q^{(i)}}) \neq 0$. This means that $\Pr(T'(w)=0) \geq \frac{1}{m}\Pr(T(w)=0)$, so T' has soundness $\frac{\mu}{m}$.

It remains to show that T' is linearly Δ-separable. To that end, observe that

$$T' = (T^{(i,j)}, (a_1^{(i,j)}, \ldots, a_q^{(i,j)}), p_{(i,j)})_{(i,j)\in I\times[m]},$$

where $p_{(i,j)} := \frac{p_i}{m}$, $a_\ell^{(i,j)} := a_\ell^{(i)}$ ($i \in I$, $j \in [m]$, $\ell \in [q]$), and $T^{(i,j)}(u)$ ($u \in \Sigma^q$) is defined to be 1 if $u \in \ker h_{ij}$ and 0 otherwise. Fix some $i \in I$ and $j \in [m]$. Then there are linear maps $g_1, \ldots, g_q : \Sigma \to \Delta$ such that $h_{ij}(u) = g_1(u_1) + \cdots + g_q(u_q)$ for all $u \in \Sigma^q$. This means that we can recover $T^{(i,j)}(u)$ from $g_1(u_1), \ldots, g_q(u_q)$, so $T^{(i,j)}$ factors via $g_1 \times \cdots \times g_q : \Sigma^q \to \Delta^q$. ◀

7 Alphabet Reduction for Good LTCs

We finally put the results of the previous sections together to prove our alphabet reduction results for good LTCs.

We begin with results about $\mathbb{F}$-codes. Theorem 3 is an immediate consequence of the following theorem applied with $(q,c) = (2,2)$ and $(q,c) = (3,1)$.

▶ **Theorem 28.** *Let $q \geq 2$ be an integer and let Δ be an $\mathbb{F}$-vector space of dimension $d > 0$. We further fix an auxiliary parameter $c \in \{1, \dots, d\}$ and require that $q \geq 3$ if $c = 1$. Suppose that $C \subseteq \Sigma^n$ is an $\mathbb{F}$-code with relative distance δ and rate r admitting an $\mathbb{F}$-linear q-query tester T with soundness $\mu > 0$. Then there exists an $\mathbb{F}$-code $C' \subseteq \Delta^{|\Sigma|^c n}$ with relative distance $(1 - |\mathbb{F}|^{-c})\delta$ and rate $\frac{1}{d} \cdot \frac{\dim \Sigma}{|\Sigma|^c} \cdot r$ admitting an $\mathbb{F}$-linear q-query tester T' with soundness*

$$\frac{c\mu\nu}{(q \dim \Sigma + c)(q|\Sigma|^c + \nu) + c\mu(1+\nu)},$$

where ν is the largest possible soundness of a q-query tester for the generalized Hadamard code $H(\Sigma, \mathbb{F}^c)$ (see § 5.2). Moreover, $\nu \geq |\Sigma|^{-2c}$ if $c > 1$ and $\nu \geq |\Sigma|^{-3}$ if $c = 1$.

Our requirement about q implies that the theorem cannot be applied with 1-dimensional Δ and $q = 2$.

Note that increasing the auxiliary parameter c decreases the rate of C', and also the ratio between the block-lengths of C and C'. On the other hand, we expect that increasing c would increase ν, and thus the soundness of T'. In particular, we believe that ν is much larger than the naive lower bound provided in the theorem. For example, when $q = 3$ and $c = 1$, we have $\nu \geq \frac{1}{6}$ by [20, Lem. B.2]; see also [21, Thm. 4.8] which shows that ν is close to 1 when $c = 1$.

Proof. Fix a c-dimensional subspace $\Delta' \subseteq \Delta$. By Theorem 27, we may assume that T is Δ'-separable at the cost of reducing the soundness from μ to $\mu' := \frac{c}{q \dim \Sigma + c}\mu$ (note that $\lceil \frac{q \dim \Sigma}{c} \rceil^{-1} \geq (\frac{q \dim \Sigma}{c} + 1)^{-1} = \frac{c}{q \dim \Sigma + c}$). Now, by Proposition 24(ii), there is an $\mathbb{F}$-linear injective function $f : \Sigma \to (\Delta')^k$ such that T is f-compatible. Moreover, by Remark 25, $D := \mathrm{im}(f)$ is the generalized Hadamard code $H(\Sigma, \Delta')$, and so $k = |\Sigma|^c$. By the definition of ν, the code $D \subseteq (\Delta')^k$ admits an $\mathbb{F}$-linear q-tester with soundness ν. Moreover, since $q \geq 2$ when $c > 1$ (resp. $q \geq 3$ with $c = 1$), ν is greater than or equal to the soundness of the 2-dependence (resp. 3-dependence) tester for $H(\Sigma, \Delta')$. The latter is positive by Theorem 16 (resp. the discussion in § 5.2), so $\nu \geq k^{-2} = |\Sigma|^{-2c}$ (resp. $\nu \geq k^{-3} = |\Sigma|^{-3}$).

Consider the code $C'' := C \circ_f D \subseteq (\Delta')^{kn}$. By Proposition 6, it has relative distance $\delta(H(\Sigma, \Delta'))\delta(C) = (1 - |\Delta'|^{-1})\delta$ and rate $r(H(\Sigma, \Delta'))r(C) = \frac{r \dim \Sigma}{c|\Sigma|^c}$, and by Theorem 8, it has an $\mathbb{F}$-linear q-tester with soundness

$$\mu'' := \frac{\mu'\nu}{qk + \mu' + \nu} = \frac{c\mu\nu}{(q \dim \Sigma + c)(q|\Sigma|^c + \nu) + c\mu}.$$

Finally, we take C' to be C'' viewed as a code with alphabet Δ (instead of Δ'). This multiplies the rate by $\frac{\dim \Delta'}{\dim \Delta} = \frac{c}{d}$, has no effect on the relative distance, and by Proposition 10, C' still admits an $\mathbb{F}$-linear q-tester with soundness $\frac{\mu''}{\mu'' + 1} = \frac{c\mu\nu}{(q \dim \Sigma + c)(q|\Sigma|^c + \nu) + c\mu(1+\nu)}$. ◀

Theorem 2 follows by applying following theorem with $(q, c) = (2, 3)$ and $(q, c) = (3, 2)$.

▶ **Theorem 29.** *Let $q \geq 2$ be an integer and let Δ be an alphabet with d elements. We further fix an auxiliary parameter $c \in \{2, \dots, d\}$ and require that $q \geq 3$ if $c = 2$. Suppose that $C \subseteq \Sigma^n$ is a code with relative distance δ and rate r admitting a possibly adaptive q-query tester T with soundness $\mu > 0$. Then there exists a code $C' \subseteq \Delta^{c^{|\Sigma|} n}$ with relative distance $(1 - c^{-1})\delta$ and rate $\frac{\log_d |\Sigma|}{c^{|\Sigma|}} \cdot r$ admitting a q-query tester T' with soundness*

$$\frac{\mu\nu}{|\Sigma|^q(qc^{|\Sigma|} + \nu) + \mu(\nu + 1)},$$

where ν is the largest possible soundness of a q-query tester for the generalized long code $L(\Sigma, \{1, \dots, c\})$ (see § 5.3). Moreover, $\nu \geq c^{-2|\Sigma|}$ if $c > 2$ and $\nu \geq 2^{-3|\Sigma|}$ if $c = 2$.

Our requirement about q implies that the theorem cannot be applied when $q = 2$ and $|\Delta| = 2$. As in Theorem 28, the parameter c allows a trade-off between code-theoretic properties of C' and the soundness of the tester T', where again, we believe that ν is much larger than the naive bounds given in the theorem.

Proof. The proof strategy is similar to that of proving Theorem 28.

Let Δ' be a subset of Δ having c elements. We use Theorem 26 to replace T with a Δ'-separable tester with soundness $\mu' := \frac{\mu}{|\Sigma|^q}$. By Proposition 23(ii), there is an injective function $f : \Sigma \to (\Delta')^k$ such that T is f-compatible, and by Remark 25, it can be chosen so that $D := \mathrm{im}(f)$ is the generalized long code $L(\Sigma, \Delta')$. In particular, $k = c^{|\Sigma|}$. By Theorem 17 (resp. Proposition 18), when $c > 2$ (resp. $c = 2$), the 2-dependence (resp. 3-dependence) for D has positive soundness. As a result, when $c > 2$ (resp. $c = 2$), D has a q-tester with soundness $\nu \geq k^{-2}$ (resp. $\nu \geq k^{-3}$).

The proof is finished as in Theorem 28 by taking C' to be the code $C \circ_f D \subseteq (\Delta')^{kn}$ and viewing it as a code inside Δ^{kn}. ◀

Next, we note that the parameters of the code C' and the tester T' promised by Theorem 29 can be improved if C is assumed to be an $\mathbb{F}_2$-code and T is $\mathbb{F}_2$-linear. This is the content of the following theorem, which we state only in the case $|\Delta| = 3$ for simplicity.

▶ **Theorem 30.** *Suppose that $C \subseteq \Sigma^n$ is an $\mathbb{F}_2$-code with relative distance δ and rate r admitting an $\mathbb{F}_2$-linear q-query tester T with soundness $\mu > 0$. Then there exists a code $C' \subseteq \{0,1,2\}^{(2|\Sigma|^2+|\Sigma|)n}$ with relative distance $\frac{\delta}{2|\Sigma|^2+|\Sigma|}$ and rate $\frac{\log_3 |\Sigma|}{2|\Sigma|^2+|\Sigma|} \cdot r$ admitting a q-query tester T' with soundness*

$$\frac{\mu\nu}{q \dim \Sigma \cdot (2q|\Sigma|^2 + q|\Sigma| + \nu) + \mu},$$

where $\nu \geq (2|\Sigma|^2 + |\Sigma|)^{-2}$ is the maximal possible soundness of a 2-query tester for a certain code $D \subseteq \{0,1,2\}^{2|\Sigma|^2+|\Sigma|}$ specified in the proof.

We will need the following lemma, whose straightforward proof may be found in the extended version of this paper [10, Lem. 7.4].

▶ **Lemma 31.** *Let $C \subseteq \Sigma^n$ be a code with a q-tester T. Let Δ be an alphabet and let Δ' be another alphabet containing Δ. Let $f_1, \ldots, f_k : \Sigma \to \Delta$ and let $f'_1, \ldots, f'_{k'} : \Sigma \to \Delta'$. Define $f : \Sigma \to \Delta^k$ by $f(a) = (f_1(a), \ldots, f_k(a))$ and define $f' : \Sigma \to (\Delta')^{k'}$ similarly. Suppose that for every $i \in [k]$, there is $i' \in [k']$ such that $f'_{i'} = f_i$ as functions from Σ to Δ'. If T is f-compatible, then it is f'-compatible.*

Proof of Theorem 30. By Theorem 27, we may replace T with a linearly $\mathbb{F}_2$-separable tester at the cost of reducing the soundness to $\mu' := \frac{\mu}{q \dim \Sigma}$.

Let $g_1, \ldots, g_t$ be all the linear $\mathbb{F}_2$-functions from Σ to $\mathbb{F}_2$ (so $t = |\Sigma|$), and let $g : \Sigma \to \{0,1\}^t$ be defined by $g(a) = (g_1(a), \ldots, g_t(a))$. By Proposition 24(ii), T is g-compatible. Now define $f_1, \ldots, f_\ell : \Sigma \to \{0,1,2\}$ as in Lemma 19 (with ℓ in place of n). By construction, $\ell \leq t + 2t^2 =: k$, and if $\ell < t + 2t^2$ we define $f_{\ell+1}, \ldots, f_k : \Sigma \to \{0,1,2\}$ to be identically 0. Now define $f : \Sigma \to \{0,1,2\}^k$ similarly to g. By Lemma 31, T is also f-compatible.

Put $D = D(\{f_i\}_{i=1}^k) \subseteq \{0,1,2\}^k$ and let $C' = C \circ_f D \subseteq \{0,1,2\}^{kn}$. By Lemma 19, $D \subseteq \{0,1,2\}^k$ has a 2-query tester with soundness $k^{-2} = (t + 2t^2)^{-2}$, so $\nu \geq (t + 2t^2)^{-2}$. Furthermore, by Theorem 8, $C' \subseteq \{0,1,2\}^{kn}$ has a 2-query tester with soundness

$$\frac{\mu'\nu}{qk + \mu' + \nu} = \frac{\mu\nu}{q \dim \Sigma \cdot (2q|\Sigma|^2 + q|\Sigma| + \nu) + \mu}.$$

To finish, note that by Proposition 6, $\delta(C') \geq \delta(D)\delta(C) \geq \frac{\delta}{k}$ and $r(C') = r(D)r(C) = \frac{\log_3 |\Sigma|}{k} \cdot r$. ◀

▶ Remark 32. By tracing the proofs of Theorems 28, 29, 30, one sees that if the tester T of C has randomness complexity R, then the promised tester T' for C' has randomness complexity $\max\{R, \log_2 n\} + O(1)$, where the $O(1)$-factor depends on μ, $|\Sigma|$, $|\Delta|$.

References

1 M. Bellare, D. Coppersmith, J. Hastad, M. Kiwi, and M. Sudan. Linearity testing in characteristic two. *IEEE Transactions on Information Theory*, 42(6):1781–1795, 1996. doi:10.1109/18.556674.

2 Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, PCPs, and nonapproximability—towards tight results. *SIAM Journal on Computing*, 27(3):804–915, 1998. doi:10.1137/S0097539796302531.

3 Eli Ben-Sasson, Oded Goldreich, and Madhu Sudan. Bounds on 2-query codeword testing. In *Approximation, randomization, and combinatorial optimization*, volume 2764 of *Lecture Notes in Comput. Sci.*, pages 216–227. Springer, Berlin, 2003. doi:10.1007/978-3-540-45198-3_19.

4 Eli Ben-Sasson, Venkatesan Guruswami, Tali Kaufman, Madhu Sudan, and Michael Viderman. Locally testable codes require redundant testers. *SIAM J. Comput.*, 39(7):3230–3247, July 2010. doi:10.1137/090779875.

5 Eli Ben-Sasson and Madhu Sudan. Robust locally testable codes and products of codes. *Random Structures Algorithms*, 28(4):387–402, 2006. doi:10.1002/rsa.20120.

6 Eli Ben-Sasson and Michael Viderman. Towards lower bounds on locally testable codes via density arguments. *Comput. Complexity*, 21(2):267–309, 2012. doi:10.1007/s00037-012-0042-8.

7 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, 1993. doi:10.1016/0022-0000(93)90044-W.

8 Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):Art. 12, 44, 2007. doi:10.1145/1236457.1236459.

9 Irit Dinur, Shai Evra, Ron Livne, Alexander Lubotzky, and Shahar Mozes. Locally testable codes with constant rate, distance, and locality. In *STOC '22—Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 357–374. ACM, New York, 2022.

10 Uriya First and Stav Lazarovici. Good locally testable codes with small alphabet and small query size. *arXiv preprint*, 2025. doi:10.48550/arXiv.2512.16082.

11 Uriya A. First and Tali Kaufman. Cosystolic expansion of sheaves on posets with applications to good 2-query locally testable codes and lifted codes. In *STOC'24—Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1446–1457. ACM, New York, 2024. doi:10.1145/3618260.3649625.

12 Uriya A. First and Tali Kaufman. Cosystolic expansion of sheaves on posets with applications to good 2-query locally testable codes and lifted codes. *arXiv preprint*, 2024. doi:10.48550/arXiv.2403.19388.

13 Katalin Friedl and Madhu Sudan. Some improvements to total degree tests. In *Third Israel Symposium on the Theory of Computing and Systems (Tel Aviv, 1995)*, pages 190–198. IEEE Comput. Soc. Press, Los Alamitos, CA, 1995. doi:10.1109/ISTCS.1995.377032.

14 Oded Goldreich. Short locally testable codes and proofs. In *Studies in complexity and cryptography*, volume 6650 of *Lecture Notes in Comput. Sci.*, pages 333–372. Springer, Heidelberg, 2011. doi:10.1007/978-3-642-22670-0_25.

15 Oded Goldreich and Madhu Sudan. Locally testable codes and PCPs of almost-linear length. *J. ACM*, 53(4):558–655, 2006. doi:10.1145/1162349.1162351.

16 Sivakanth Gopi, Swastik Kopparty, Rafael Oliveira, Noga Ron-Zewi, and Shubhangi Saraf. Locally testable and locally correctable codes approaching the Gilbert-Varshamov bound. *IEEE Trans. Inform. Theory*, 64(8):5813–5831, 2018. doi:10.1109/TIT.2018.2809788.

17 Venkatesan Guruswami, Atri Rudra, and Madhu Sudan. Essential coding theory, 2025. Preprint. Aug 26, 2025 version. Available at `https://cse.buffalo.edu/faculty/atri/courses/coding-theory/book/`.

18 Gillat Kol and Ran Raz. Bounds on 2-query locally testable codes with affine tests. *Inform. Process. Lett.*, 116(8):521–525, 2016. `doi:10.1016/j.ipl.2016.03.008`.

19 Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally correctable and locally testable codes with sub-polynomial query complexity. *J. ACM*, 64(2):Art. 11, 42, 2017. `doi:10.1145/3051093`.

20 Or Meir. Combinatorial construction of locally testable codes. *SIAM J. Comput.*, 39(2):491–544, 2009. `doi:10.1137/080729967`.

21 Tushant Mittal and Sourya Roy. A general framework for low soundness homomorphism testing. In *17th Innovations in Theoretical Computer Science Conference*, volume 362 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 103, 18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. `doi:10.4230/lipics.itcs.2026.103`.

22 Pavel Panteleev and Gleb Kalachev. Asymptotically good quantum and locally testable classical LDPC codes. In *STOC '22 – Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 375–388. ACM, New York, 2022.

23 Alexander Polishchuk and Daniel A. Spielman. Nearly-linear size holographic proofs. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '94, pages 194–203, New York, NY, USA, 1994. Association for Computing Machinery. `doi:10.1145/195058.195132`.

24 Ron Roth. *Introduction to Coding Theory*. Cambridge University Press, 2006.

25 Michael Viderman. Strong LTCs with inverse polylogarithmic rate and soundness. In *2013 IEEE Conference on Computational Complexity – CCC 2013*, pages 255–265. IEEE Computer Soc., Los Alamitos, CA, 2013. `doi:10.1109/CCC.2013.34`.

Parallel Sampling from the Ising p-Spin Model

Nima Anari
Stanford University, CA, USA

Aniket Das
Stanford University, CA, USA

Alireza Haqi
Stanford University, CA, USA

Abstract

We study the parallel complexity of sampling from the high-temperature Ising mixed p-spin Gibbs measure, a canonical instance of a mean-field spin glass on the hypercube $\{\pm 1\}^n$. We propose two different algorithms for this problem, corresponding to two different regimes of accuracy.

Our first algorithm is a parallel implementation of a Markov chain known as block dynamics, combined with an approximate rejection sampling step that uses an Ising model in a novel way as a proposal distribution to approximate the quadratic interaction terms of the p-spin Hamiltonian. For any $\varepsilon > 0$, this algorithm runs in $n^{1/3}\mathrm{polylog}(n/\varepsilon)$ parallel time with $\mathrm{poly}(n/\varepsilon)$ work, and outputs a sample whose law is ε-close to the p-spin measure in total variation distance.

Our second algorithm uses Picard iterations to parallelize the Algorithmic Stochastic Localization (ASL) process of El Alaoui, Montanari, and Sellke (2025), and for any $\varepsilon > \varepsilon_n$, takes $\mathrm{polylog}(n/\varepsilon)$ parallel time and $\mathrm{poly}(n/\varepsilon)$ work to produce a sample that is ε-close to the p-spin measure in the normalized 2-Wasserstein metric. Here, $\varepsilon_n > 0$ is a threshold that goes to 0 as $n \to \infty$. Our result constitutes a doubly exponential improvement in the ε dependence of the runtime and an exponential improvement in the ε dependence of the total work when compared to naïve ASL, whose runtime scales as $\exp(\mathrm{poly}(1/\varepsilon))$.

2012 ACM Subject Classification Theory of computation → Parallel algorithms; Theory of computation → Random walks and Markov chains; Mathematics of computing → Probabilistic algorithms

Keywords and phrases spin glasses, parallel sampling, Glauber dynamics, stochastic localization

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.71

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2607.12348

Funding This work is supported by funds from NSF-2125511, NSF CCF-2045354, NSF MAI-2501597.

1 Introduction

Efficient sampling from a probability distribution which is specified only up to an unknown normalizing constant is a central algorithmic problem across computer science, statistical physics, and machine learning. Generally, given a finite state space Ω and a Hamiltonian $H : \Omega \to \mathbb{R}$, the target distribution is a Gibbs measure of the following form:

$$\mu(x) = \frac{1}{Z}\exp(H(x)), \qquad Z = \sum_{x\in\Omega}\exp(H(x)) \tag{1}$$

where the normalizing constant Z, also known as the partition function, is unknown and typically hard to compute. This setting includes classical problems in approximate counting [32] and randomized algorithms such as approximating the volume of convex bodies [15] and approximating matrix permanents [31], Markov-chain Monte Carlo methods in statistical physics [41], and posterior sampling in high-dimensional statistical models.

Our work studies the problem of sampling from the Gibbs measure of the Ising mixed p-spin model, which is a Gibbs measure on the hypercube $\{\pm 1\}^n$ whose Hamiltonian is a random polynomial with Gaussian coefficients.

© Nima Anari, Aniket Das, and Alireza Haqi;
licensed under Creative Commons License CC-BY 4.0

$$H(x) = \sum_{p=2}^{P} \frac{\beta_p \sqrt{p!}}{n^{\frac{p-1}{2}}} \cdot \sum_{J \subseteq [n], |J|=p} G_J \prod_{k \in J} x_k + \langle x, h \rangle, \qquad \mu(x) = \frac{1}{Z} \exp(H(x)) \tag{2}$$

Here, $P \geq 2$ is an integer, the disorder $G_J \overset{i.i.d}{\sim} \mathcal{N}(0,1)$ for every choice of J, and h is an arbitrary external field independent of G. The p-spin Hamiltonian can be equivalently characterized as a Gaussian process on $\{\pm 1\}^n$ with $\mathbb{E}_G[H(x)] = \langle h, x \rangle$ and $\mathrm{Cov}[H(x)H(y)] = N\xi(\langle x,y \rangle / N)$, where $\xi(t) = \sum_{p=2}^{P} \beta_p^2 t^p$ denotes the mixture function. The mixed p-spin model is a canonical instance of a mean-field spin glass, which plays a central role in several areas of probability, mathematical physics, and average-case complexity [37, 46, 40, 38]. Notable special cases include the pure p-spin model where only one β_p is nonzero, and the Sherrington-Kirkpatrick (SK) model where $P = 2$.

We consider the problem of approximate sampling from the p-spin Gibbs measure in the quenched setting, where the disorder G is sampled once and then fixed, and the algorithm is tasked with approximately sampling from μ given random access to the disorder coefficients (which can be stored with $\mathrm{poly}(n)$ memory). Given an error parameter $\varepsilon > 0$, the goal is to design an algorithm that runs in time $\mathrm{poly}(n, 1/\varepsilon)$ for any realization of G, and produces an output distributed as $\hat{\mu}$ such that $\mathrm{dist}(\hat{\mu}, \mu) \leq \varepsilon$ for typical realizations of G. Here, dist denotes any appropriate metric between probability distributions.

Under this setting, [1] and [7] analyze Glauber dynamics [27], a canonical Markov chain for sampling from discrete distributions, which, at every step, resamples a randomly chosen coordinate according to its conditional marginal, while fixing the remaining coordinates. The authors identify a high-temperature regime for the p-spin model under which Glauber dynamics efficiently samples from μ. Specifically, they show that when $\sum_{p=2}^{P} \beta_p \sqrt{p^3 \ln(p)} \leq C$ for some small enough $C = \Theta(1)$, Glauber dynamics takes $O(n \ln(n/\varepsilon))$ time to output a sample distributed as $\hat{\mu}$ satisfying $d_{\mathrm{TV}}(\hat{\mu}, \mu) \leq \varepsilon$ with high probability over the randomness of G. Since the error parameter $\varepsilon > 0$ can be made as small as desired, we refer to such a guarantee as a *High Accuracy Guarantee*.

In another line of work, [19] propose a different approximate sampler for the p-spin model with a zero external field (i.e., $h = 0$ in Equation (2)) based on an approximate discretization of an Itô diffusion process known as Eldan's stochastic localization [20]. Their algorithm, named Algorithmic Stochastic Localization (ASL), offers a weaker sampling guarantee, albeit supporting a broader range of temperatures than that covered by the high-temperature condition defined above. Concretely, there exists an error threshold $\varepsilon_n = o_n(1)$ such that for any $\varepsilon > \varepsilon_n$, ASL runs in time $O(\mathrm{poly}(n) \exp(\mathrm{poly}(1/\varepsilon)))$ and with high probability over G, approximately samples from μ up to error ε in the normalized 2-Wasserstein metric. In other words, the output $\hat{x}$ of their algorithm can be coupled with some $x^\star \sim \mu$ such that $n^{-1}\,\mathbb{E}[\|\hat{x} - x^\star\|] \leq \varepsilon^2$. We call such a guarantee a *Low Accuracy Guarantee* for two key reasons. Firstly, for a fixed problem instance, the accuracy of ASL can only be certified above a threshold $\varepsilon > \varepsilon_n$. Secondly, their guarantees are presented in the normalized 2-Wasserstein metric, which is strictly weaker than the total variation metric for the range of ε tolerated by their algorithm.

A common drawback of both algorithms considered above is that they are inherently sequential. Indeed, Glauber dynamics sequentially updates one coordinate after another in a random order, while Algorithmic Stochastic Localization discretizes the trajectory of a continuous Markov process, which naturally evolves in a sequential fashion. Consequently, these algorithms fail to utilize the full potential of modern hardware, which is generally capable of massive parallelism, both in terms of parallel computation and parallel memory access. This motivates the central problem of our work:

Can we design an efficient approximate sampling algorithm for the Ising p-spin model that can take advantage of parallel hardware?

Specifically, we consider the problem of sampling from the p-spin Gibbs measure in the PRAM model of computation, where the algorithm is allowed to query the disorder G in multiple rounds, with each round comprising polynomially many synchronous queries that are executed in parallel. Given an error tolerance $\varepsilon > 0$, our goal is to design an algorithm that approximately samples from μ up to error ε in some suitable metric for typical realizations of G, while also optimizing for two criteria: *Parallel Runtime* or *Round Complexity*, which denotes the number of rounds of parallel queries made to G, and *Work*, which denotes the total number of queries made to G. Ideally, for every realization of G, we want the parallel runtime to scale at most sublinearly in n and ε^{-1}, and the work to scale at most polynomially in n and ε^{-1}.

Compared to the complexity of sequential sampling, the theory of parallel sampling for mean-field spin glasses is far less developed. To our knowledge, the only prior parallel algorithm for sampling from the p-spin Gibbs measure is the work of [34], which considers the p-spin model in the high-temperature regime and achieves a high-accuracy guarantee with a parallel runtime of $O(\sqrt{n} \cdot \mathrm{polylog}(n/\varepsilon))$ and $O(\mathrm{poly}(n/\varepsilon))$ work. Beyond this, for the special case of the high-temperature SK model (i.e., $P = 2$ and $\beta_2 < 1/4$), [11] show that the Restricted Gaussian Dynamics algorithm of [12] is a high-accuracy approximate parallel sampler with parallel runtime $O(\mathrm{polylog}(n/\varepsilon))$ and $O(\mathrm{poly}(n/\varepsilon))$ work.

1.1 Our Contributions

In this work, we develop two different algorithms for parallel sampling from the Ising mixed p-spin Gibbs measure. The first is a high-accuracy algorithm which samples from the p-spin model up to ε total variation error in $O(n^{1/3}\mathrm{polylog}(n/\varepsilon))$ parallel time and $O(\mathrm{poly}(n, \log(n/\varepsilon)))$ work. To our knowledge, this constitutes an improvement both in terms of parallel runtime and work compared to the previous best known result of [34], which exhibits a parallel runtime of $O(\sqrt{n}\mathrm{polylog}(n/\varepsilon))$ with $O(\mathrm{poly}(n/\varepsilon))$ work.

▶ **Theorem 1** (High-Accuracy Parallel p-spin Sampler, Informal Version of Theorem 7)**.** *Let μ denote the Ising p-spin Gibbs measure as defined in Equation (2). Assume the model is in the high-temperature regime, i.e., $\sum_{p=2}^{P} \beta_p \sqrt{p^3 \ln(p)} \le C$ for some $C = \Theta(1)$. Then, there exists a parallel sampling algorithm (Algorithm 1) satisfying the following for any $\varepsilon > 0$.*

- *The algorithm runs in parallel time $\widetilde{O}(n^{1/3} \ln(1/\varepsilon))$ with $O(\mathrm{poly}(n, \log(n/\varepsilon)))$ work.*
- *With probability at least $1 - \exp(-cn)$ over the randomness of the disorder G, the algorithm outputs $\hat{x} \sim \hat{\mu}$ such that $d_{\mathrm{TV}}(\hat{\mu}, \mu) \le \varepsilon$.*

Our second algorithm obtains a low-accuracy approximate sampling guarantee in the normalized 2-Wasserstein metric, and enjoys a parallel runtime of $O(\mathrm{polylog}(n/\varepsilon))$ with $O(\mathrm{poly}(n/\varepsilon))$ work. This significantly improves upon the $O(\mathrm{poly}(n) \cdot \exp(\mathrm{poly}(1/\varepsilon)))$ runtime of [19] and constitutes the first parallel algorithm for sampling from the p-spin model with $\mathrm{polylog}(n/\varepsilon)$ parallel runtime in any metric.

▶ **Theorem 2** (Low-Accuracy Parallel p-spin Sampler, Informal Version of Theorem 16)**.** *Let μ denote the Ising p-spin Gibbs measure as defined in Equation (2) with $h = 0$. Assume the model is in the high-temperature regime, i.e., $\sum_{p=2}^{P} \beta_p \sqrt{p^3 \ln(p)} \le C$ for some $C = \Theta(1)$. Then, there exists an $\varepsilon_n > 0$ such that $\varepsilon_n \to 0$ as $n \to \infty$, and a parallel sampling algorithm (Algorithm 4) which satisfies the following for any $\varepsilon > \varepsilon_n$:*

- *The algorithm runs in parallel time $O(\mathrm{polylog}(n/\varepsilon))$ with $O(\mathrm{poly}(n/\varepsilon))$ work.*
- *With probability at least $1 - o(1)$ over the randomness of the disorder G, the output $\hat{x}$ of the algorithm can be coupled to some $x^\star \sim \mu$ such that $n^{-1}\,\mathbb{E}[\|\hat{x} - x^\star\|^2] \le \varepsilon^2$.*

For the special case of the SK model, our low-accuracy algorithm satisfies the above guarantee for $\beta_2 \leq 0.375$. To the best of our knowledge, this constitutes the only parallel sampler for the SK model running in $O(\mathrm{polylog}(n/\varepsilon))$ time with $O(\mathrm{poly}(n/\varepsilon))$ work in this regime of inverse temperatures.

1.2 Related Work

1.2.1 Sampling from mean-field spin glasses

For the Sherrington–Kirkpatrick model ($P = 2$), classical mixing criteria such as Dobrushin's condition [14] imply $\widetilde{O}(n^2)$ mixing time of Glauber dynamics only up to the vanishing window $\beta_2 = O(n^{-1/2})$. This falls short of the dimension-free β_2 regime predicted in the statistical physics literature and leads to algorithmically unsatisfactory guarantees. The first major breakthrough was the work of [21] which used stochastic localization to prove that Glauber dynamics mixes in $\widetilde{O}(n^2)$ time up to $\beta_2 < 1/4$. The mixing time was subsequently sharpened to $\widetilde{O}(n)$ by [6] and [12], and the range of β_2 was improved to $\beta_2 < 0.295$ by [8]. For the Ising p-spin model, [1] proved an $\widetilde{O}(n^2)$ mixing time for Glauber dynamics in the high-temperature regime $\sum_{2\leq p\leq P} \beta_p \sqrt{p^3 \ln(p)} = O(1)$. The mixing time was later improved to $\widetilde{O}(n)$ by [7].

In a different direction, [17] (see also [10]) introduced Algorithmic Stochastic Localization (ASL) for the SK model with zero external field. For any $\varepsilon > \varepsilon_n$ for a fixed $\varepsilon_n = o_n(1)$, their algorithm runs in time $O(\mathrm{poly}(n)\exp(\mathrm{poly}(1/\varepsilon)))$ and obtains a low-accuracy guarantee: sampling up to ε-error in the normalized 2-Wasserstein distance. Their algorithm works up to $\beta_2 < 1$, which is conjectured to be the sharp threshold up to which the SK model admits a polynomial-time approximate sampler. Their framework was later extended to the Ising p-spin model in [19], obtaining a similar low-accuracy guarantee but covering a broader range of temperatures than that of [1] and [7].

There also exists a parallel line of work on the spherical p-spin model, where the Gibbs measure μ is supported on the sphere $\{x \in \mathbb{R}^n \mid \|x\|^2 = n\}$, a setting that is often more analytically tractable than the hypercube. In the high-temperature regime, [26] proved that a continuous-time diffusion process known as Langevin dynamics samples from the spherical p-spin model in total variation in $\widetilde{O}(1)$ time. Beyond the high-temperature regime considered in [26], the recent breakthrough of [30] designed an improved version of ASL for the spherical p-spin model with zero external field that samples up to $o(1)$ error in total variation in $O(\mathrm{poly}(n))$ time beyond the high-temperature regime considered by [26]. Building upon their techniques, [29] proved that continuous-time annealed Langevin dynamics samples up to total variation error $\exp(-\Omega(n^{-1/5}))$ in $O(\mathrm{poly}(n))$ time beyond the high-temperature regime.

1.2.2 Parallel sampling

Parallel sampling has been a longstanding topic of interest in theoretical computer science, and more recently, in machine learning, where diffusion-based generative models are often deployed on massively parallel hardware [33, 44, 28]. In theoretical computer science, the problem dates back to at least [39], who developed an RNC algorithm (i.e., running in $O(\mathrm{polylog}(n))$ parallel time on a PRAM with $O(\mathrm{poly}(n))$ work) for finding a perfect matching, and asked whether one can also sample a uniform random perfect matching in parallel; this remains a major open problem to date.

Recent works study parallel sampling assuming access to global counting oracles. One such model is the weighted counting oracle, where one has access to the log Laplace transform $L(w) = \ln \mathbb{E}_\mu[\exp(\langle w, x\rangle)]$. A weaker model is the unweighted counting oracle, which returns

conditional marginals $\mathbb{P}_{X\sim\mu}[X_i = a \mid X_S = \omega_S]$ for $S \subseteq [n]$, $i \notin S$, $a \in \{\pm 1\}$, and $\omega_S \in \{\pm 1\}^S$. Such models are of interest for several combinatorial distributions such as spanning trees, Eulerian tours, planar perfect matchings, and determinantal point processes, where such oracles can be implemented on a PRAM in $O(\text{polylog}(n))$ time and $O(\text{poly}(n))$ work [13]. Assuming access to weighted counting oracles, [5, 3] obtained RNC samplers for distributions satisfying $\|\nabla^2 L(w)\|_{\text{op}} = O(1)$. For more general distributions that do not satisfy such a regularity condition, [28] developed an algorithm running in $\widetilde{O}(n^{2/3})$ parallel time with polynomial work. The parallel runtime was later improved to $\widetilde{O}(\sqrt{n})$ by [2]. In the weaker unweighted counting model, [4] developed an $O(n^{2/3})$ time parallel sampler with $O(\text{poly}(n))$ work. The parallel runtime was later improved to $O(\sqrt{n})$ by [2].

Contrary to the above, our work falls under the more challenging local oracle setting, where one has to design a parallel sampler for a Gibbs measure given access to its Hamiltonian. This constitutes a far weaker form of access compared to the global counting oracles described above. In fact, the problem of computing a counting oracle of a Gibbs measure given its Hamiltonian is NP-hard in the worst case [25], and highly nontrivial even for specialized Hamiltonians like the SK model. In this setting, [23, 36] develop RNC samplers for Ising models satisfying a Dobrushin-type condition. For mean-field spin glass models, this only covers a vanishing window of temperatures, e.g., $\beta_2 = O(n^{-1/2})$ for the SK model. More recently, [11] show that the Restricted Gaussian Dynamics algorithm of [12] is an RNC sampler for the SK model. For general Ising p-spin models, much less is known. To our knowledge, the only parallel sampler in this setting prior to our work is [34], which runs in $\widetilde{O}(\sqrt{n})$ parallel time with polynomial work.

1.3 Overview of Our Techniques

Algorithm 1 s-Glauber Dynamics with Parallel Rejection Sampling.

Sample $X^{(0)}$ from $\mu_0 \propto e^{\langle h,x\rangle}$
for $t = 0, \ldots, T-1$ **do**
  Sample $S \sim \text{Uniform}\binom{[n]}{s}$
  $X_S^{(t+1)} \leftarrow \textsc{ParallelRejectionSampler}(S, X_{\overline{S}}^{(t)}, \varepsilon/10T)$ *(see Algorithm 2)*
  $X_{\overline{S}}^{(t+1)} \leftarrow X_{\overline{S}}^{(t)}$
return $X^{(T)}$

1.3.1 High Accuracy Sampler

Motivated by the prior work of [34], our high-accuracy sampler for the p-spin model, stated in Algorithm 1, is a computationally efficient implementation of a Markov chain called s-Glauber dynamics (also known as s-block dynamics), whose kernel $P_s(X^{(t)}, \cdot)$ is defined as follows:

1. Sample S uniformly at random from all subsets of $[n]$ of size s.
2. Sample $X^{(t+1)}$ from the posterior $\mu\left(X^{(t+1)} \;\middle|\; X_S^{(t+1)} = X_S^{(t)}\right)$

The above Markov chain is reversible with respect to μ and, for $s = 1$, it corresponds to Glauber dynamics, a canonical sampling algorithm for discrete spaces. When μ is the p-spin Gibbs measure at high temperature, s-Glauber dynamics is known to exhibit a mixing time of $\widetilde{O}(n/s)$ [7, 34]. Although this mixing time improves upon increasing s, the computational complexity of implementing each step of s-Glauber dynamics worsens significantly as s grows.

Algorithm 2 Approximate Parallel Rejection Sampling with an Ising Proposal (ParallelRejectionSampler).

Input: S, pinning τ on $\overline{S}$, $\varepsilon_{\text{step}}$

Set $y = X_S$ and decompose the conditional Hamiltonian $H_{S,\tau}(y) = H(y, \tau)$ on S as follows

$$H(y) = \langle b_{S,\tau}, y\rangle + \frac{1}{2}\, y^\top A_{S,\tau} y + R_{S,\tau}(y).$$

Set the rejection parameter $\bar{c} = \Theta(1)$, $L = \Theta(\bar{c}\ln(1/\varepsilon_{\text{step}}))$, $\varepsilon_{\text{RGD}} = \Theta(\varepsilon_{\text{step}}/L)$, $\eta = \Theta(1)$, and $\delta = \Theta(\varepsilon_{\text{step}})$

$\hat{h} = \textsc{CenteringExternalField}(A_{S,\tau}, b_{S,\tau}, R_{S,\tau}, \eta, \delta)$ *(see Algorithm 6)*

for $\ell = 1$ **to** L **do In Parallel**

- $U_\ell, V_\ell \overset{\text{iid}}{\sim} \text{RGD}\Big(A_{S,\tau},\, b_{S,\tau} + \hat{h},\, \varepsilon_{\text{RGD}}\Big)$
- $W_\ell \sim \mathsf{Uniform}[0,1]$
- Accept U_ℓ if $W_\ell \le \min\big\{1, \bar{c}^{-1}\exp(\psi(U_\ell) - \psi(V_\ell))\big\}$ where $\psi(y) = R_{S,\tau}(y) - \Big\langle \hat{h}, y \Big\rangle$

return *the first accepted sample; declare FAILURE otherwise*

Indeed, a naïve implementation of the posterior sampling step (i.e., Step 2 above) takes $O(2^s)$ time, which is computationally prohibitive, and completely negates the potential benefits of any mixing time improvements for $s > 1$.

To circumvent this, our Algorithm 1 replaces the posterior sampling step in s-Glauber dynamics with an approximate rejection sampler (Algorithm 2), whose proposal distribution is a carefully chosen Ising model. The idea of using rejection sampling in the posterior sampling step also appears in the work of [34], which approximates the posterior sampling step with a product proposal. While this enables computational efficiency (as product measures can be sampled in $O(1)$ parallel time), their rejection sampling procedure fails to correctly approximate the posterior when $s = \omega(\sqrt{n})$, and obtains a parallel runtime of $\widetilde{O}(n^{1/2})$. In contrast, the choice of our Ising proposal, which serves as the key technical innovation underlying our parallel speedup, can be made to approximate the posterior to arbitrary accuracy as long as $s = O(n^{2/3})$, leading to an improved parallel runtime of $\widetilde{O}(n^{1/3})$.

To motivate the construction of our Ising proposal, we observe that the conditional posterior distribution $\mu_{S,\tau}(y) \coloneqq \mu(x_S = y \mid x_{\bar{S}} = \tau) \propto \exp(H_{S,\tau}(y))$ is also a p-spin Gibbs measure whose Hamiltonian, $H_{S,\tau}(y) = H(x)$ for $x_S = y$ and $x_{\bar{S}} = \tau$, is obtained by restricting H to the coordinates in S and pinning the coordinates in $\bar{S}$ according to τ. $H_{S,\tau}$ is then a polynomial of the underlying disorder G, and admits the following decomposition:

$$H_{S,\tau}(y) = \langle b_{S,\tau}, y\rangle + \frac{1}{2}\langle y, A_{S,\tau} y\rangle + R_{S,\tau}(y) \tag{3}$$

where $b_{S,\tau}$, $A_{S,\tau}$, and $R_{S,\tau}$ depend on G. Motivated by this decomposition, our rejection sampler uses a proposal distribution $\pi_{S,\tau}$ such that $\ln \pi_{S,\tau}$ serves as a quadratic approximation to $H_{S,\tau}$, i.e., $\pi_{S,\tau}(x) \propto \exp\Big(\frac{1}{2}\langle y, A_{S,\tau} y\rangle + \Big\langle b_{S,\tau} + \hat{h}, y\Big\rangle\Big)$[1]. The performance and accuracy of our algorithm are governed by two key problems: 1. How easily can we sample from the proposal? 2. How well does our rejection sampler approximate the posterior? Below, we discuss how we address each of these questions in our proof of Theorem 1.

[1] As explained in Section 3.2, the nonzero external field $\hat{h}$ *centers* our rejection sampling proposal, leading to improved accuracy.

Analyzing the parallel complexity of our algorithm is a delicate problem due to the choice of our proposal distribution. Indeed, sampling from an Ising model is in itself a nontrivial problem, which can even be NP-hard in the worst case [45, 24]. To resolve this, we use the concentration properties of the p-spin Hamiltonian to prove that sampling from $\pi_{S,\tau}$ is computationally tractable. In particular, the Hamiltonian of $\pi_{S,\tau}$ satisfies a spectral condition which ensures that a Markov chain known as Restricted Gaussian Dynamics (or RGD, see Algorithm 3) approximately samples from $\pi_{S,\tau}$ in total variation, with $\widetilde{O}(1)$ mixing time [12]. Moreover, each step of RGD can be implemented in $\widetilde{O}(1)$ parallel time with $O(\mathrm{poly}(n))$ work [11]. Thus, to ensure parallel efficiency, our rejection sampler uses approximate samples from the proposal distribution, drawn via $\widetilde{O}(1)$ parallel calls to RGD, thereby implying an $\widetilde{O}(1)$ parallel runtime for Algorithm 2, which in turn leads to an $\widetilde{O}(n/s)$ parallel runtime for Algorithm 1.

Algorithm 3 Restricted Gaussian Dynamics (RGD) [12, 11].

Input: symmetric interaction matrix $J \in \mathbb{R}^{m\times m}$, external field $h \in \mathbb{R}^m$, tolerance ε
Output: Sample X satisfying $d_{\mathsf{TV}}(\mathsf{Law}(X), \nu_{J,h}) \le \epsilon$ where
$\nu_{J,h}(x) \propto \exp(\frac{1}{2}\langle x, Jx\rangle + \langle h, x\rangle)$
Initialize $X^{(0)} \in \{\pm 1\}^m$ arbitrarily
Set $T = \Theta(\log(m/\epsilon))$
for $t = 1, \dots, T$ **do**
  Sample $y \sim \mathcal{N}(X^{(t)}, J^{-1})$
  Sample $X^{(t)}$ from the product measure $p(x) \propto \exp(\langle x, h + Jy\rangle)$
return $X^{(T)}$

Finally, the optimal choice of s is determined by the approximation error of our rejection sampler. There are two main sources of this error, namely, the approximate sampling error of RGD (which can be controlled to arbitrary precision without sacrificing algorithmic efficiency), and the error due to mismatch between the proposal $\pi_{S,\tau}$ and the conditional posterior $\mu_{S,\tau}$, which is governed by the log-density $\psi(y) = \ln \frac{\mathrm{d}\mu_{S,\tau}}{\mathrm{d}\pi_{S,\tau}}(y)$. We show that this error can be controlled by proving exponential tail bounds on ψ under $\pi_{S,\tau}$, which we establish via the concentration properties of the p-spin Hamiltonian and the isoperimetric properties of the proposal, and conclude that Algorithm 2 uniformly approximates the posterior sampling step of s-Glauber dynamics for $s \le O(n^{2/3})$, leading to an overall runtime of $\widetilde{O}(n^{1/3})$ for Algorithm 1.

1.3.2 Low Accuracy Sampler

Adapting the framework of [17, 19], our low-accuracy parallel sampler, presented in Algorithm 4, is based on Eldan's Stochastic Localization (SL) [20], which we briefly introduce. For any probability measure μ supported on $\{\pm 1\}^n$, the associated SL process $(y_t)_{t\ge 0}$ is defined by the following stochastic differential equation on $\mathbb{R}^n$.

$$\mathrm{d}y_t = m(y_t)\,\mathrm{d}t + \mathrm{d}B_t, \quad y_0 = 0, \qquad m(y) = \frac{\mathbb{E}_{x\sim\mu}[x\cdot\exp(\langle y, x\rangle)]}{\mathbb{E}_{x\sim\mu}[\exp(\langle y, x\rangle)]} \tag{4}$$

Here, $(B_t)_{t\ge 0}$ denotes the standard Brownian motion on $\mathbb{R}^n$, and $m(y)$ corresponds to the mean of the exponential tilt of the measure μ, which we call the tilted mean. A key property of SL is that $t^{-1}y_t$ is marginally distributed as $x^\star + \zeta_t$ where $x^\star \sim \mu$ and

$\zeta_t \sim \mathcal{N}(0, t^{-1}I)$. Intuitively, $t^{-1}y_t$ is approximately distributed as μ for large enough t, and thus, a suitable discretization of Equation (4), such as the Euler discretization below, can be used to approximately sample from μ.

$$\widetilde{y}_{(k+1)h} = \widetilde{y}_{kh} + hm(\widetilde{y}_{kh}) + (B_{(k+1)h} - B_{kh}),\ 0 \le k < K, \qquad y_0 = 0 \tag{5}$$

Being the Euler discretization of a continuous Markov process, the discrete dynamical system specified by Equation (5) naturally evolves in a sequential fashion, and thus, naively simulating K steps requires $\Theta(K)$ calls to an oracle for the tilted mean m, even on a PRAM. To circumvent this, we employ a parallelization technique based on the concept of Picard iteration, a classical technique for proving the existence and uniqueness of ODE solutions [42], which was later used algorithmically by [3] to develop parallel algorithms for sampling from continuous log-concave densities in $\mathbb{R}^n$. The key idea behind Picard iteration is to view the trajectory defined by Equation (5) as the solution to the following fixed point problem defined by an operator $\mathcal{T} : \mathbb{R}^{n\times K} \to \mathbb{R}^{n\times K}$ over discrete trajectories:

$$\mathcal{T}(z) \coloneqq z'; \qquad z'_{kh} = h\sum_{i=0}^{k-1} m(z_{ih}) \ + B_{kh}\ \forall\ k \in [K], \qquad z'_0 = 0 \tag{6}$$

By unrolling the recurrence, one can observe that the trajectory defined by Equation (5) is a fixed point of the operator $\mathcal{T}$, i.e., $\mathcal{T}(\widetilde{y}) = \widetilde{y}$. Consequently, the discretized SL process can be approximated via the following fixed point iteration over discrete trajectories, which we call Picard iteration.

$$\widetilde{y}^{(r)}_{kh} = h\sum_{i=0}^{k-1} m(\widetilde{y}^{(r-1)}_{ih}) + B_{kh},\ \forall\ k \in [K], \qquad \widetilde{y}^{(r)}_0 = 0 \tag{7}$$

Algorithm 4 Picard Algorithmic Stochastic Localization.

Input : Parallel depth R, steps K, step size h, parameters $(\eta, K_{\mathrm{AMP}}, K_{\mathrm{NGD}})$
$B_0 \leftarrow 0$
for $k \leftarrow 0$ **to** $K-1$ **do In Parallel**
  $B_{(k+1)h} \leftarrow B_{kh} + \varepsilon_k$, where $\varepsilon_k \sim \mathcal{N}(0, hI)$
  $\hat{y}^{(0)}_{kh} \leftarrow 0$
for $r \leftarrow 1$ **to** R **do**
  $\hat{y}^{(r)}_0 \leftarrow 0$
  for $k \leftarrow 0$ **to** $K-1$ **do In Parallel**
    $\hat{m}\big(\hat{y}^{(r-1)}_{kh}\big) \leftarrow \text{TAP-AMP}\big(\hat{y}^{(r-1)}_{kh}, \eta, q_\star(kh), K_{\mathrm{AMP}}, K_{\mathrm{NGD}}\big)$
      (see Algorithm 5) and Equation (18)
  for $k \leftarrow 1$ **to** K **do In Parallel**
    $\hat{y}^{(r)}_{kh} \leftarrow h\sum_{j=0}^{k-1} \hat{m}\big(\hat{y}^{(r-1)}_{jh}\big) + B_{kh}$
Output : $\hat{x} \leftarrow \mathrm{sign}\big(\hat{y}^{(R)}_{Kh}\big)$

The key algorithmic benefit of this reformulation lies in its inherent parallelizability, albeit at the cost of introducing some mild approximation error. Indeed, Equation (7) turns the problem of sequentially evaluating the recurrence in Equation (5) into a highly parallelizable fixed point computation over the whole trajectory: given access to a parallel oracle for the tilted mean m, each Picard iteration can be implemented efficiently in parallel across all time

points. In particular, at depth r, $(\widetilde{y}_{kh}^{(r-1)})_{k\in[K]}$ can be computed simultaneously in one parallel oracle call, after which the cumulative drift terms in Equation (7) can be computed via parallel prefix sums in $O(\log(k))$ time and $O(k)$ work. However, to guarantee the effectiveness of this approach as a parallel sampler, one needs to quantify how well the Picard iterate $\widetilde{y}^{(R)}$ approximates its limiting trajectory $\widetilde{y}$. To this end, one can show (see Lemma 6) that Lipschitzness of the tilted mean (which can be guaranteed in the high-temperature regime) suffices to ensure that the Picard iterates converge to the discretized SL process exponentially fast. In particular, $R = \widetilde{\Theta}(Kh)$ iterations suffice to approximate the discrete SL trajectory in Equation (5) to arbitrary accuracy.

The key challenge in algorithmically implementing the Picard iteration in Equation (7) arises from the approximate computation of the tilted mean m, which is highly nontrivial and is known to be NP-hard even for worst-case Hamiltonians [25]. To circumvent this, we make use of the TAP-AMP algorithm of [18] (Algorithm 5), which constructs an estimator $\hat{m}$ of the tilted mean for points along the continuous SL trajectory. This results in the following inexact fixed point iteration, which we call Picard Algorithmic Stochastic Localization (Algorithm 4).

$$\hat{y}_{kh}^{(r)} = h\sum_{i=0}^{k-1}\hat{m}(\hat{y}_{ih}^{(r-1)}) + B_{kh},\ \forall\ k\in[K],\quad \hat{y}_0^{(r)} = 0 \tag{8}$$

While standard perturbation-based analyses of inexact fixed point iterations rely on uniformly controlling the deviation between the ideal and the inexact iterates, such a strategy is not applicable to Equation (8) due to the absence of uniform guarantees on the estimation error of $\hat{m}$. In fact, the current best known analysis of the tilted mean estimator can certify only a very weak error bound of the form $\|\hat{m}(y_t) - m(y_t)\|^2 \le n\varepsilon_n^2$ for points along the continuous SL trajectory. Here, ε_n is a fixed error threshold which converges to 0 as $n\to\infty$. This subtle distinction constitutes a major obstacle in our analysis since the Picard trajectories at low depth (i.e., low values of r) are expected to deviate significantly from the continuous SL trajectory. Consequently, the currently available guarantees are insufficient for controlling $\|\hat{m}(\hat{y}_{kh}^{(r)}) - m(\hat{y}_{kh}^{(r)})\|$.

To circumvent this obstacle, we depart from the standard perturbation-based analysis of inexact fixed point iteration, and instead analyze Equation (8) directly. In particular, we prove that in the high-temperature regime, the tilted mean estimator $\hat{m}(y)$ itself is uniformly $O(1)$-Lipschitz. Hence, the inexact Picard iteration in Equation (8) converges exponentially fast to the discrete process $\hat{y}_{(k+1)h} = \hat{y}_{kh} + h\hat{m}(y_{kh}) + (B_{(k+1)h} - B_{kh})$ in $R = \widetilde{O}(Kh)$ iterations. We then prove a sign rounding stability guarantee for the limiting discrete trajectory $\hat{y}$, and show that for any $\varepsilon \ge \varepsilon_n$, the distribution of $\mathrm{sign}(\hat{y}_{Kh}^{(R)}/Kh)$ approximates the p-spin measure μ up to ε error in normalized 2-Wasserstein distance whenever $Kh = O(\log(n/\varepsilon))$ and $h = O(\mathrm{poly}(1/\varepsilon))$. Since the K iterations are executed in parallel in each round, Algorithm 4 exhibits a parallel runtime of $O(\mathrm{polylog}(n/\varepsilon))$ with $O(\mathrm{poly}(n/\varepsilon))$ work. Compared to the $O(\mathrm{poly}(n)\exp(\mathrm{poly}(1/\varepsilon)))$ runtime of Algorithmic Stochastic Localization, this constitutes a doubly exponential improvement in runtime and an exponential improvement in work in terms of ε.

2 Preliminaries

2.1 Notation

For differentiable functions $g:\mathbb{R}^d\to\mathbb{R}$, we use ∇g and $\nabla^2 g$ to denote their Euclidean gradient and Hessian, respectively. For functions $f:\{\pm1\}^m\to\mathbb{R}$, we define the discrete partial derivative, and the associated discrete gradient and discrete Hessian as follows:

Algorithm 5 AMP Algorithm for Tilted Mean Estimation (TAP-AMP) [19, Algorithm 1].

Input : Iterations $K_{\mathrm{AMP}}, K_{\mathrm{NGD}}$, step size η, overlap parameter q

$\hat{m}^{-1} \leftarrow 0$, $z^0 \leftarrow 0$

for $k \leftarrow 0$ **to** $K_{\mathrm{AMP}} - 1$ **do**

$\hat{m}^k \leftarrow \tanh(z^k)$

$\hat{q}_k \leftarrow \frac{1}{n}\sum_{i=1}^n \tanh^2(z_{i,k})$

$b_k \leftarrow (1-\hat{q}_k)\,\xi''(\hat{q}_k)$

$z^{k+1} \leftarrow \beta \nabla H(\hat{m}^k) + y - b_k \hat{m}^{k-1}$

$u^0 \leftarrow z^{K_{\mathrm{AMP}}}$, $\hat{m}^{+,0} \leftarrow \hat{m}^{K_{\mathrm{AMP}}}$

for $k \leftarrow 0$ **to** $K_{\mathrm{NGD}} - 1$ **do**

$u^{k+1} \leftarrow u^k - \eta \nabla \hat{\mathcal{F}}_{\mathrm{TAP}}(\hat{m}^{+,k}; y, q)$ *(see Equation (22))*

$\hat{m}^{+,k+1} \leftarrow \tanh(u^{k+1})$

Output : $\hat{m}^{+,K_{\mathrm{NGD}}}$

$$\partial_i f(x) = \frac{1}{2}\cdot[f(x_{\neg i}\ ;\ x_i = +1) - f(x_{\neg i}\ ;\ x_i = -1)],$$
$$(\nabla f(x))_i = \partial_i f(x), \qquad (\nabla^2 f(x))_{ij} = \partial_i \partial_j f(x) \tag{9}$$

For any two distributions μ and ν we use $d_{\mathsf{TV}}(\mu,\nu)$ to denote their total variation distance. When μ and ν are distributions on $\mathbb{R}^d$, we use $W_2(\mu,\nu)$ to denote their Euclidean 2-Wasserstein distance and $W_{2,d}(\mu,\nu) = d^{-1/2}W_2(\mu,\nu)$ to denote its normalized variant. Throughout, we use H and μ to denote the Ising p-spin Hamiltonian and its associated Gibbs measure as defined in Equation (2). We use $\xi(t) = \sum_{p=2}^P \beta_p^2 t^p$ to denote its mixture function. We also define the coefficients $\mathfrak{C}(\beta)$ and $\mathfrak{D}(\beta)$ as follows:

$$\mathfrak{C}(\beta) \coloneqq \sum_{p=2}^P \beta_p \sqrt{p^3 \ln(p)}\ ; \qquad \mathfrak{D}(\beta) \coloneqq \sum_{p=2}^P \beta_p \sqrt{2^p p^3 \ln(p)} \tag{10}$$

2.2 Stochastic Localization

Let μ denote any arbitrary measure on $\mathbb{R}^n$ with a finite second moment. Stochastic localization, introduced by [20], denotes the following diffusion process:

$$\mathrm{d}y_t = m(y_t,t)\,\mathrm{d}t + \mathrm{d}B_t, \qquad y_0 = 0$$
$$m(y,t) = \mathbb{E}_{x\sim\mu,\zeta\sim\mathcal{N}(0,1)}\left[x \;\middle|\; tx + \sqrt{t}\zeta = y\right] \tag{11}$$

Stochastic localization exhibits several interesting properties that have found innumerable applications in areas such as convex geometry and the analysis of Markov chains [35, 21, 12]. Among these, the following property, which appears in [16], will be of particular importance to us.

▶ **Theorem 3** (Equivalent Characterization of SL [16]). *Let $(y_t)_{t\geq 0}$ denote the stochastic localization process associated with the measure μ, as defined in Equation (11). Then, there exists an $x^\star \sim \mu$ and a standard Brownian motion $(W_t)_{t\geq 0}$ such that for any $t \geq 0$, $y_t = tx^\star + W_t$. Consequently, the marginal law of $t^{-1}y_t$ is equal to that of $x^\star + \zeta_t$ where $\zeta_t \sim \mathcal{N}(0, t^{-1}I)$.*

The following lemmas, which follow from Theorem 3, are vital for analyzing the rounding scheme used in Algorithm 4.

▶ **Lemma 4** (Rounding SL for the hypercube). *Let μ be a measure supported on $\{\pm 1\}^n$, and let $(y_t)_{t\geq 0}$ denote the associated SL process. For any $T > 0$, define $\hat{x}_T = \mathrm{sign}(T^{-1}y_T)$. Then, $W_{2,n}^2(\mathsf{Law}(\hat{x}_T), \mu) \leq 8\exp(-T/2)$.*

▶ **Lemma 5** (Stability of SL Rounding). *Let μ be a distribution on $\{\pm 1\}^n$, and let $(y_t)_{t\geq 0}$ denote the associated SL process. For any $T > 0$, let $\hat{z}_T$ be a random variable satisfying $W_2^2(\hat{z}_T, T^{-1}y_T) \leq n\Delta^2$. Then,*

$$W_{2,n}^2\big(\mathrm{sign}(\hat{z}_T), \mathrm{sign}(T^{-1}y_T)\big) \leq 8e^{-T/8} + 16\Delta^2. \tag{12}$$

2.3 Picard Iteration

Consider a discrete-time dynamical system $(x_{kh})_{0\leq k\leq K}$ on $\mathbb{R}^n$ defined by the following iteration:

$$x_{(k+1)h} = x_{kh} + hf(kh, x_{kh}) + g(kh), \qquad k \in \{0, \ldots, K-1\} \tag{13}$$

where $h > 0$ denotes the step-size and $T = Kh$ denotes the time horizon, and the functions f and g can potentially be random. We observe that solving the difference equation above is an inherently sequential task. Concretely, given an initialization x_0 and access to an oracle for computing f and g, computing the trajectory $(x_{kh})_{0\leq k\leq K}$ requires $\Theta(K)$ oracle calls.

The key idea in Picard iteration is to view the entire trajectory $(x_{kh})_{0\leq k\leq K}$ as the fixed point of a trajectory-level iteration in $\mathbb{R}^{n\times K}$ such that, given access to a *parallel oracle* for computing f and g, each round of this trajectory-level iteration can be computed in $\Theta(1)$ oracle calls. To begin, we unroll the recurrence in Equation (13) and observe that the solution $(x_{kh})_{0\leq k\leq K}$ satisfies the following:

$$x_{kh} = x_0 + \sum_{j=0}^{k-1} \left(hf(jh, x_{jh}) + g(jh)\right), \qquad 0 \leq k \leq K \tag{14}$$

The Picard iteration associated with Equation (13) is the following sequence of trajectories $(x_{kh}^{(r)})_{0\leq k\leq K},\ r \in \mathbb{N}$.

$$x_{kh}^{(r+1)} = x_0 + \sum_{j=0}^{k-1} \left(hf(jh, x_{jh}^{(r)}) + g(jh)\right) \tag{15}$$

We make two key observations regarding the above trajectory sequence:

- The solution x_{kh} to Equation (13) is a fixed point of Equation (15). Indeed, setting $x_{kh}^{(r)} = x_{kh}$ for every k and using Equation (14), we conclude that $x_{kh}^{(r+1)} = x_{kh}$.
- Each round of the iteration in Equation (15) can be computed using $\Theta(1)$ calls to a parallel oracle for computing f and g. Indeed, the values of $g(jh),\ 0 \leq j < K$ can be precomputed using one parallel oracle call, and in each round, the values of $f(jh, x_{jh}^{(r)}),\ 0 \leq j < K$ can be computed via one call to a parallel oracle to f.

The parallel efficiency of Picard iteration then depends on how fast the sequence of trajectories in Picard iteration converges to its fixed point trajectory. Below, we prove that Picard iteration exhibits an exponential convergence rate whenever $f(t, x)$ is uniformly L-Lipschitz in x.

▶ **Lemma 6** (Exponential convergence of Picard iteration under Lipschitzness). *Let $(x_{kh})_{0\le k\le K}$ denote the solution to Equation (13), where $T = Kh$, and let $(x_{kh}^{(r)})_{0\le k\le K,\ r\ge 0}$ denote the associated Picard iteration as defined in Equation (15). Suppose $f(t,\cdot)$ is uniformly L-Lipschitz in x and define $M = \max_{0\le k\le K}\|x_{kh}^{(0)} - x_{kh}\|$. Then, for any $r \ge 0$ and $0 \le k \le K$,*

$$\|x_{kh}^{(r)} - x_{kh}\| \le \frac{(khL)^r}{r!}\cdot M. \tag{16}$$

In particular, $R = \max\{e^2 LT, \ln(M/\varepsilon)\}$ Picard iterations suffice to ensure the following:

$$\max_{k\in\{0,\dots,K\}} \|x_{kh}^{(R)} - x_{kh}\| \le \varepsilon. \tag{17}$$

2.4 TAP Free Energy and the Tilted Mean Estimator

The construction of the tilted mean estimator (Algorithm 5) of El Alaoui, Montanari, and Sellke [19] is motivated by a deep structural property of the p-spin model: The mean of the tilted Gibbs measure at high temperature is an approximate stationary point of the Thouless–Anderson–Palmer (TAP) free energy [47, 37, 46], which we briefly introduce below.

Recall that the mixture function of the p-spin model is defined as $\xi(t) = \sum_{p=2}^{P}\beta_p^2 t^p$. For any $t \ge 0$, the asymptotic overlap $q^\star(t)$ is defined as follows:

$$q_{k+1}(t) = \mathbb{E}_{W\sim N(0,1)}\left[\tanh\left(t + \xi'(q_k) + W\sqrt{\xi'(q_k)+t}\right)\right], \quad q_k(t) = 0, \quad q^\star(t) = \lim_{k\to\infty} q_k(t) \tag{18}$$

Since $q_k(t)$ is independent of G, computable to high accuracy via a one-dimensional Gaussian integral, and approaches the limit $q^\star(t)$ exponentially fast, we assume, for simplicity, that $q^\star(t)$ is known exactly for $t = kh, k \in [K]$.

The TAP free energy associated with the p-spin Hamiltonian in Equation (2) is defined as follows:

$$\mathcal{F}_{\mathrm{TAP}}(m, y) = -H(m) - \langle y, m\rangle - \sum_{i=1}^{n} h(m_i) - \mathrm{ONS}(Q(m)) \tag{19}$$

$$\mathrm{ONS}(q) = \frac{n}{2}\left[\xi(1) - \xi(Q) - (1-Q)\xi'(Q)\right] \tag{20}$$

$$Q(m) = n^{-1}\|m\|^2, \qquad h(m) = -\frac{1+m}{2}\ln\left(\frac{1+m}{2}\right) - \frac{1-m}{2}\ln\left(\frac{1-m}{2}\right) \tag{21}$$

For the sake of tractability, El Alaoui, Montanari, and Sellke [19] also define an approximate version of the TAP free energy which replaces the $\mathrm{ONS}(Q(m))$ term by a quadratic approximation.

$$\hat{\mathcal{F}}_{\mathrm{TAP}}(m\,; y, q) = -H(m) - \langle y, m\rangle - \sum_{i=1}^{n} h(m_i) - \mathrm{ONS}(q) - \mathrm{ONS}'(q)(Q(m) - q) + \frac{n\Gamma}{8}(Q(m) - q)^2 \tag{22}$$

where q is set to be the asymptotic overlap $q^\star(t)$ for $t = kh$, and $\Gamma = \Theta(1)$ is a tunable parameter depending only on ξ. [19] show that the mean of the tilted Gibbs measure continues to be an approximate stationary point of $\hat{\mathcal{F}}_{\mathrm{TAP}}$. The tilted mean estimator in Algorithm 5 then works in two phases. The first phase runs a series of AMP updates to compute an approximate stationary point $\hat{m}^{K_{\mathrm{AMP}}}$ for $\hat{\mathcal{F}}_{\mathrm{TAP}}$ around which it is locally strongly convex. The second stage exploits this local strong convexity to refine the AMP estimate via natural gradient descent.

3 High Accuracy Parallel Sampler for the p-Spin Model

In this section, we analyze Algorithm 1 and prove the following Theorem.

▶ **Theorem 7** (Analysis of Algorithm 1). *For any $\varepsilon > 0$, the output $X^{(T)}$ of Algorithm 1 run with $s = \Theta(n^{2/3}\log(n/\varepsilon)^{-2/3})$ and $T = \Theta(n^{1/3}\log(n/\varepsilon)^{5/3})$ satisfies $d_{\mathsf{TV}}(\mathsf{Law}(X^{(T)}), \mu) \le \varepsilon$ with probability at least $1 - e^{-cn}$. Moreover, Algorithm 1 has a parallel runtime of $O(n^{1/3}\mathrm{polylog}(n/\varepsilon))$ with $O(\mathrm{poly}(n, \log(n/\varepsilon)))$ work.*

We describe the key technical ideas of our proof below, and refer the readers to the full version of our paper for complete proofs.

3.1 Bounds on the Hamiltonian of the Conditional Posterior

Consider any arbitrary $S \in \binom{[n]}{s}$. For any $x \in \{\pm 1\}^n$, let $x = (y, \tau)$ where $y = x_S$ denotes the block spins in S and $\tau = x_{\bar{S}}$ denotes the pinning outside S. Then the conditional distribution $\mu_{S,\tau}(y) = \mu(y \mid x_{\bar{S}} = \tau)$ of the spins in S satisfies $\mu_{S,\tau}(y) \propto \exp(H_{S,\tau}(y))$ where:

$$H_{S,\tau}(y) = \langle b_{S,\tau}, y\rangle + \tfrac{1}{2}\langle y, A_{S,\tau}y\rangle + R_{S,\tau}(y) \tag{23}$$

Here, $A_{S,\tau}$ is a symmetric matrix with zero diagonals with:

$$(A_{S,\tau})_{ij} = \sum_{p=2}^{P} \frac{\beta_p\sqrt{p!}}{n^{\frac{p-1}{2}}} \sum_{K\subseteq\bar{S},|K|=p-2} G_{K\cup\{i,j\}}\tau^K, \qquad i,j \in S,\ i \ne j \tag{24}$$

The remainder term is given by

$$R_{S,\tau}(y) = \sum_{p=3}^{P} \frac{\beta_p\sqrt{p!}}{n^{(p-1)/2}} \sum_{\substack{J\subseteq[n],|J|=p,\\ |J\cap S|\ge 3}} G_J\ y^{J\cap S}\ \tau^{J\cap\bar{S}} \tag{25}$$

As discussed earlier, each step of s-Glauber dynamics requires sampling from $\mu_{S,\tau}$, which can be computationally expensive. We address this via a parallelized implementation of rejection sampling in Algorithm 2 where $\mu_{S,\tau}(y)$ is approximated by an Ising model of the form $\pi_{S,\tau}(y) \propto \exp(\frac{1}{2}\langle y, A_{S,\tau}y\rangle + \left\langle b_{S,\tau} + \hat{h}, y\right\rangle)$, where $\hat{h}$ is an external field that satisfies $\hat{h} \approx \mathbb{E}_{y\sim\pi_{S,\tau}}[\nabla R_{S,\tau}(y)]$.

To ensure the parallel efficiency of Algorithm 2, we generate approximate samples from the Ising proposal $\pi_{S,\tau}$ using Restricted Gaussian Dynamics (Algorithm 3). To this end, the following theorem uses the concentration properties of the p-spin Hamiltonian to prove that $\|A_{S,\tau}\|_{\mathsf{op}} \le 1/4$ holds with high probability uniformly over all choices of S and τ whenever $s = O(n^{2/3})$. Consequently, Algorithm 3 can approximately sample from the proposal in $\widetilde{O}(1)$ parallel time.

▶ **Theorem 8** (Uniform Operator Norm Bound for Ising Proposal). *There exists a numerical constant $C > 0$ such that for $\mathfrak{C}(\beta) \le C$ and $s = o(n)$, the following holds with probability at least $1 - \exp(-cn)$:*

$$\|A_{S,\tau}\|_{\mathsf{op}} \le \frac{1}{4}, \qquad \forall\ S \in \binom{[n]}{s},\ \tau \in \{\pm 1\}^{\bar{S}} \tag{26}$$

Naturally, the success probability of our rejection sampling procedure is governed by how well the Ising proposal $\pi_{S,\tau}$ approximates the conditional distribution $\mu_{S,\tau}$, or, in other words,

how large the influence of the remainder term $R_{S,\tau}$ is. We quantify this via a uniform bound on the discrete Hessian of $R_{S,\tau}$, which follows from the concentration properties of the p-spin Hamiltonian and a careful combinatorial bookkeeping.

▶ **Theorem 9** (Uniform Bound on Hessian of Remainder Term). *There exists a numerical constant $C > 0$ such that for $\mathfrak{C}(\beta) \le C$, the following holds with probability at least $1 - \exp(-cn)$ whenever $\Omega(\sqrt{n}) \le s \le o(n)$*

$$\|\nabla^2 R_{S,\tau}(y)\|_F^2 \le \frac{C_1 \mathfrak{C}(\beta)^2 s^3}{n^2}, \qquad \forall\ S \in \binom{[n]}{s},\ (y,\tau) \in \{\pm 1\}^n \tag{27}$$

3.2 Analysis of Parallel Rejection Sampling

In this section, we shall establish the accuracy and parallel efficiency of Algorithm 2. Henceforth, we condition on the event $\mathcal{E}$ that the bounds in Theorem 8 and Theorem 9 hold uniformly over all S and τ.

▶ **Theorem 10** (Analysis of Parallel Rejection Sampler). *For any $S \in \binom{[n]}{s}$ and $\tau \in \{\pm 1\}^{\overline{S}}$, let $\tilde{\mu}_{S,\tau}$ denote the law of the sample output by Algorithm 2. Conditioned on the event $\mathcal{E}$, the following holds whenever $s = \Theta(n^{2/3})$:*

$$d_{\mathsf{TV}}(\tilde{\mu}_{S,\tau}, \mu_{S,\tau}) \le \varepsilon_{\mathrm{step}} \qquad \forall S \in \binom{[n]}{s},\ \tau \in \{\pm 1\}^{\overline{S}}. \tag{28}$$

Moreover, Algorithm 2 has parallel runtime $O(\mathrm{polylog}(n/\varepsilon_{\mathrm{step}}))$ with $O(\mathrm{poly}(n, \log(1/\varepsilon_{\mathrm{step}})))$ work.

The proof of Theorem 10, whose complete proof is relegated to the full version, involves several technical components, the first of which is an accuracy guarantee for the approximate rejection sampling step in Algorithm 2. This result appears as Lemma 4.4 in Lee [34] and Lemma 2 in [22].

▶ **Lemma 11** (Accuracy of Rejection Sampling). *Let P and Q be probability measures satisfying $\frac{dP}{dQ}(y) \propto e^{\psi(y)}$. Let $Y, Z \overset{\text{iid}}{\sim} Q$, and define $R = \exp(\psi(Y) - \psi(Z))$. Consider the following rejection sampling procedure: Draw $U \sim \mathsf{Uniform}[0,1]$ and accept Y if $U \le \min\{1, R/c\}$ for some $c \ge 1$. Let $\tilde{P}$ denote the law of the accepted sample and assume $\psi(Y) - \psi(Z)$ exhibits subexponential tails of the form $\mathbb{P}[\psi(Y) - \psi(Z) \ge t] \le a\exp(-bt)$ for $b > 1$. Then, $d_{\mathsf{TV}}\left(P, \tilde{P}\right) \le \frac{a}{b-1} \cdot c^{-(b-1)}$*

In Algorithm 2, (X, Z) corresponds to (U_ℓ, V_ℓ), both of which are approximately distributed as $\pi_{S,\tau}$ (modulo the negligible sampling error of Algorithm 3, which we ignore for now). Since $\frac{d\mu_{S,\tau}}{d\pi_{S,\tau}}(y) \propto e^{\psi(y)}$, where $\psi(y) = R_{S,\tau}(y) - \langle \hat{h}, y\rangle$, Lemma 11 suggests that we can bound the TV error of Algorithm 2 by controlling the sampling error of Algorithm 3 and proving a subexponential tail bound for $|\psi(\widetilde{U}) - \psi(\widetilde{V})|$, where $\widetilde{U}, \widetilde{V} \overset{\text{iid}}{\sim} \pi_{S,\tau}$, or equivalently, for $|\psi(Y) - \mathbb{E}[\psi(Y)]|$ under $Y \sim \pi_{S,\tau}$. By Theorem 8 and [34, Theorem 4.1], $\pi_{S,\tau}$ satisfies approximate tensorization of entropy with high probability. The work of [43] then implies the following concentration bound:

$$\mathbb{P}_{Y \sim \pi_{S,\tau}}\left[|\psi(Y) - \mathbb{E}_{\pi_{S,\tau}}[\psi(Y)]|\right] \le 2\exp\left(-c\min\left\{\frac{t^2}{\mathbb{E}_{\pi_{S,\tau}}[\|\nabla\psi\|^2]}, \frac{t}{\max\limits_{y\in\{\pm1\}^S}\|\nabla^2\psi(y)\|_F}\right\}\right) \tag{29}$$

Algorithm 6 Centering External Field.

Input: $A_{S,\tau}$, $b_{S,\tau}$, $R_{S,\tau}$, accuracy $\eta > 0$, failure probability $\delta \in (0,1)$
Set iterations $T = \Theta(\log(n/\eta))$ and sample size $K = \Theta\big(s\eta^{-2}\log(sT/\delta)\big)$
Initialize $z_0 \leftarrow 0$
for $t = 0, \ldots, T-1$ **do**
  for $i = 1, \ldots, K$ **do** In parallel
    $U_i^{(t)} \sim \mathrm{RGD}(A_{S,\tau}, b_{S,\tau} + z_t, \delta/100KT)$
  $z_{t+1} = \frac{1}{K}\sum_{i=1}^{K} \nabla R_{S,\tau}(U_i^{(t)})$
return $\hat{h}_{S,\tau} = z_T$

Since $\|\nabla^2\psi(y)\|_F^2 = \|\nabla^2 R_{S,\tau}(y)\|_F \lesssim \mathfrak{C}(\beta)\cdot\frac{s^3}{n^2}$ holds whp uniformly for any S, τ and y by Theorem 9. Hence, setting $s = \Theta(n^{2/3})$ and $\mathfrak{C}(\beta)$ sufficiently small, $\max_{y\in\{\pm1\}^S} \|\nabla^2\psi(y)\|_F \le \delta$ for any $\delta = O(1)$. To control $\mathbb{E}_{\pi_{S,\tau}}[\|\nabla\psi\|^2]$, we note that:

$$\mathbb{E}_{\pi_{S,\tau}}[\|\nabla\psi\|^2] = \| \hat{h} - \mathbb{E}_{\pi_{S,\tau}}[\nabla R_{S,\tau}] \|^2 + \mathbb{E}_{\pi_{S,\tau}}\left[\| \nabla R_{S,\tau} - \mathbb{E}[\nabla R_{S,\tau}] \|^2\right] \tag{30}$$

Using the Poincare Inequality for $\pi_{S,\tau}$ and Theorem 9, the second term in Equation (30) can be made $O(\delta^2)$ for any $\delta = O(1)$ by setting $s = \Theta(n^{2/3})$ and $\mathfrak{C}(\beta)$ sufficiently small. The centering field $\hat{h}$ must be chosen so as to make the first term $\|\hat{h} - \mathbb{E}_{\pi_{S,\tau}}[\nabla R_{S,\tau}]\|^2$ as small as possible. Since $\pi_{S,\tau} = \nu_{A_{S,\tau}, b_{S,\tau}+\hat{h}}$ where $\nu_{J,v}(y) \propto \exp(\frac{1}{2}\langle y, Jy\rangle + \langle v, y\rangle)$, the optimal choice of $\hat{h}$ is given by the solution to the following fixed point problem:

$$h = \mathbb{E}_{\nu_{A_{S,\tau},\, h+b_{S,\tau}}}[\nabla R_{S,\tau}]. \tag{31}$$

As we shall demonstrate, Equation (31) admits a unique solution which can be efficiently approximated by an inexact fixed point iteration, namely Algorithm 6, which replaces the expectation in Equation (31) by an empirical average over approximate samples drawn via RGD. In particular, we prove the following guarantee.

▶ **Theorem 12** (Analysis of Algorithm 6). *Conditioned on the event $\mathcal{E}$, the output $\hat{h}$ of Algorithm 6 satisfies $\|\hat{h} - \mathbb{E}_{\pi_{S,\tau}}[\nabla R_{S,\tau}]\| \le \eta$ with probability at least $1-\delta$. Moreover, Algorithm 6 runs in $O(\mathrm{polylog}(n, \eta^{-1}, \delta^{-1}))$ parallel time with $O\big(\mathrm{poly}(\frac{n}{\eta}, \ln(1/\delta))\big)$ work.*

The proof of Theorem 10, which is presented in our full version, then follows by combining Theorem 12 and Lemma 11

3.3 Proof of Theorem 7

Proof. Let $\mu_t = \mathsf{Law}(X^{(t)})$. Since $\mu_0(x) \propto \exp(\langle h, x\rangle)$, by Lemma 5.5 of [34],

$$\mathbb{P}\big[D_{\mathsf{KL}}(\mu_0 \,||\, \mu) \le Cn\big] \ge 1 - \exp(-cn). \tag{32}$$

Let $\mathcal{G} = \mathcal{E} \cap \big\{D_{\mathsf{KL}}(\mu_0 \,||\, \mu) \le Cn\big\}$. Then, $\mathbb{P}[\mathcal{G}] \ge 1 - e^{-cn}$, and henceforth, we condition on $\mathcal{G}$.

Let P_s and $\widetilde{P}_s$ denote the Markov kernel for s-Glauber dynamics and the Markov chain in Algorithm 1 respectively. By Theorem 10, for $s = \Theta((\frac{n}{\log(n/\varepsilon)})^{2/3})$,

$$\max_{x\in\{\pm1\}^n} d_{\mathsf{TV}}\big(P_s(x,\cdot), \widetilde{P}_s(x,\cdot)\big) \le \varepsilon_{\mathrm{step}} = \tfrac{\varepsilon}{10T}. \tag{33}$$

Hence,

$$d_{\mathsf{TV}}\left(\mu_T, \mu_0 P_s^T\right) \le T\varepsilon_{\text{step}} \le \tfrac{\varepsilon}{10}. \tag{34}$$

By [7, Theorem 6], the following holds with probability $1 - e^{-cn}$:

$$D_{\mathsf{KL}}\left(\mu_0 P_s^T \mid\mid \mu\right) \le \exp\left(-\tfrac{csT}{n}\right) Cn \le \exp\left(-\frac{cT}{n^{1/3}\log(n/\varepsilon)^{2/3}}\right) Cn. \tag{35}$$

Setting $T = \Theta(n^{1/3}\log(n/\varepsilon)^{5/3})$, applying Pinsker's inequality and taking appropriate union bounds, we conclude that the following holds with probability at least $1 - \exp(-cn)$:

$$d_{\mathsf{TV}}(\mu_T, \mu) \le d_{\mathsf{TV}}\left(\mu_T, \mu_0 P_s^T\right) + d_{\mathsf{TV}}\left(\mu_0 P_s^T, \mu\right) \le \tfrac{\varepsilon}{10} + \tfrac{\varepsilon}{2} \le \varepsilon. \tag{36}$$

To compute the parallel runtime, note that Algorithm 1 makes T calls to Algorithm 2 each of which has parallel runtime $O(\mathsf{polylog}(n/\varepsilon))$ with $O(\mathsf{poly}(n/\varepsilon))$ processors as per Theorem 10. Hence, Algorithm 1 exhibits a parallel runtime of $O(n^{1/3}\mathsf{polylog}(n/\varepsilon))$ with $O(\mathsf{poly}(n/\varepsilon))$ processors. ◀

4 Low Accuracy Parallel Sampler for the p-Spin Model

In this section, we key technical ideas of our proof below and refer the readers to the full version of our paper for complete proofs. We prove that Picard Algorithmic Stochastic Localization (Algorithm 4) samples from the mixed p-spin model with Wasserstein guarantees.

We first illustrate our results for the SK model. The TAP-AMP mean-estimation algorithm for the SK model (Algorithm 7) is slightly different from that for the mixed p-spin model, but it uses the same AMP iterations followed by NGD iterations. We prove that the TAP-AMP algorithms proposed in prior work for computing mean approximations in the SK and mixed p-spin models, Algorithm 7 and Algorithm 5, are Lipschitz with respect to the external field and $\widetilde{O}(1)$ parallel computable. This enables us to apply Picard iteration to the steps of algorithmic stochastic localization Equation (8), yielding a parallel speedup.

For the rest of the section, we first present an improved parameter dependence for algorithmic stochastic localization in Section 4.1. Then, using the improved parameters, we prove the parallelization result for algorithmic stochastic localization in Section 4.2, relying on the following two results:

- The TAP-AMP mean approximations are executable in $\widetilde{O}(1)$ parallel time (Section 4.3).
- The TAP-AMP mean approximations are Lipschitz (Section 4.4).

4.1 Sign Rounding for Algorithmic Stochastic Localization

El Alaoui, Montanari, and Sellke [17, 19] provide Wasserstein guarantees for algorithmic stochastic localization using TAP fixed points as mean approximations. Their analysis controls the propagation of the mean-estimation error along the discretized stochastic localization trajectory. They obtain an ε upper bound in normalized Wasserstein distance between the output of the sampling algorithm and the target distribution by using $\exp(\mathrm{poly}(1/\varepsilon))$ discrete steps to run the stochastic localization up to a time horizon $\mathrm{poly}(1/\varepsilon)$.

▶ **Proposition 13.** *For $\varepsilon_n < \varepsilon$, El Alaoui, Montanari, and Sellke [17, 19] provide algorithmic stochastic localization for a time horizon* $\mathrm{poly}(1/\varepsilon)$ *with* $\exp(\mathrm{poly}(1/\varepsilon))$ *discretization steps.*

In this work, in addition to the Picard parallelization and Lipschitz analysis, we improve the dependence of the discretization step size and the time horizon in stochastic localization on the parameter ε by analyzing sign rounding for algorithmic stochastic localization.

▶ **Theorem 14.** *For any n, there exists a value ε_n such that, for any $\varepsilon > \varepsilon_n$ and any high-temperature mixed p-spin model with Hamiltonian H_n and no external field, there is an algorithm based on algorithmic stochastic localization using TAP-AMP mean approximations (Algorithm 7 and Algorithm 5) whose output is within ε of the target distribution in normalized Wasserstein distance with probability $1 - o(1)$ over the disorder.*

Moreover, the algorithmic stochastic localization procedure uses a discretization step of size δ and time horizon t satisfying

$$\delta = \mathrm{poly}(\varepsilon), \qquad t = O(\log(1/\varepsilon)).$$

The output of the algorithm is generated by sign rounding, namely $\mathrm{sign}(\hat{y}_t)$, where $\hat{y}_t$ is the output of the discrete algorithmic stochastic localization process at time t. Furthermore, if y_t denotes the true stochastic localization process, then

$$W_{2,n}(y_t, \hat{y}_t) \le O(\varepsilon).$$

4.2 Parallel Time Analysis of Algorithm 4

In this section, we analyze the parallelization result for algorithmic stochastic localization using Picard iteration. The Picard iterations produce a trajectory that is close, in Wasserstein distance, to the true solution of the discrete stochastic localization process using the TAP-AMP mean approximation. This discrete process is itself close to the true continuous-time stochastic localization process by Theorem 14. Therefore, by the triangle inequality for Wasserstein distance, the input to the rounding step is close to the output of the true stochastic localization process.

We present our results for the SK model and the mixed p-spin model separately. For the SK model, the Gibbs measure is defined by

$$\mu_A(x) \propto \exp\left(\frac{\beta}{2}\langle x, Ax\rangle\right),$$

where $A \sim \mathrm{GOE}(n)$. Here $\mathrm{GOE}(n)$ denotes the distribution over symmetric matrices whose entries are Gaussian with variance of order $1/n$. Although the proof structure is nearly the same, we separate the two cases because, for the SK model, we obtain a parallel sampling algorithm in a temperature regime not covered by previous parallel algorithms. The analysis for the SK model yields the threshold $\beta_0 \approx 0.3753$ for Lipschitzness and parallelization, improving on the $\beta < 1/4$ threshold at which the previous $\mathrm{polylog}(n)$-depth algorithm for sampling from the SK model applies [11]. However, unlike the algorithm of Chen, Liu, Yin, and Zhang [11], our low-accuracy algorithm cannot achieve arbitrarily small error.

▶ **Theorem 15** (SK Model Parallelization)**.** *For any $\varepsilon \ge \varepsilon_n$ and inverse temperature $\beta < \beta_0 \approx 0.3753$, there exist parameters (Theorem 14)*

$$\eta, K_{\mathrm{AMP}}, K_{\mathrm{NGD}} = \mathrm{polylog}(n/\varepsilon), \quad K = \mathrm{poly}(1/\varepsilon), \quad \delta = \mathrm{poly}(\varepsilon), \quad t = K\delta = O(\log(1/\varepsilon))$$

such that the following holds. Let $R = O(\frac{\log^2(1/\varepsilon)}{\beta_0 - \beta})$, and execute Algorithm 4 on a Hamiltonian of the form $H_n(x) = x^\top Ax$ using TAP-AMP (Algorithm 7) as the approximate mean function $\hat{m}$ with parameters $(\eta, K_{\mathrm{AMP}}, K_{\mathrm{NGD}}, K, \delta)$.

Then Algorithm 4 outputs a random point $\mathbf{x}^{\mathrm{alg}} \in \{-1, +1\}^n$ with law $\mu_{\mathbf{A}}^{\mathrm{alg}}$ such that, with probability $1 - o(1)$ over $\mathbf{A} \sim \mathrm{GOE}(n)$,

$$W_{2,n}(\mu_{\mathbf{A}}^{\mathrm{alg}}, \mu_{\mathbf{A}}) \le O(\varepsilon).$$

The parallel runtime of the algorithm is $\mathrm{poly}(\log(n/\varepsilon))$, and its total work is $\mathrm{poly}(n/\varepsilon)$.

▶ **Theorem 16** (p-spin Model Parallelization). *For any $\varepsilon \geq \varepsilon_n$ and temperature coefficients $\{\beta_p\}_{p\geq 2}$ satisfying*

$$\mathfrak{C}(\beta) \coloneqq \sum_{p=2}^{P} \beta_p \sqrt{p^3 \ln(p)} < \gamma_0 \qquad \mathfrak{D}(\beta) = \sum_{p=2}^{P} \beta_p \sqrt{2^p p^3 \ln(p)} < \infty$$

there exist parameters (Theorem 14)

$$\eta, K_{\mathrm{AMP}}, K_{\mathrm{NGD}}, K = \mathrm{poly}(1/\varepsilon), \qquad \delta = \mathrm{poly}(\varepsilon), \qquad t = K\delta = O(\log(1/\varepsilon)),$$

such that the following holds. Let $R = O_{\mathfrak{C}(\beta)}\big(\log^2(1/\varepsilon)\big)$ be the number of Picard iterations. Given query access to the p-spin Hamiltonian as defined in Equation (2), Algorithm 4, run with parameters $(\eta, K_{\mathrm{AMP}}, K_{\mathrm{NGD}}, K, \delta)$, outputs a random point $\mathbf{x}^{\mathrm{alg}} \in \{-1, +1\}^n$ with law $\mu_{H_n}^{\mathrm{alg}}$ such that, with probability $1 - o(1)$ over the Gaussian coefficients $\{g_J\}_{J\subseteq[n]}$,

$$W_{2,n}(\mu_{H_n}^{\mathrm{alg}}, \mu_{H_n}) \leq O(\varepsilon).$$

Here, ε is the error incurred by the sequential algorithm in Theorem 14, while $\mathrm{poly}(\varepsilon)$ is the additional error incurred by replacing the sequential updates with Picard iteration.

The parallel runtime of the algorithm is $\mathrm{poly}(\log(n/\varepsilon))$ and its total work is $\mathrm{poly}(n, 1/\varepsilon)$.

4.3 TAP-AMP in $\mathrm{polylog}(n)$ Parallel Time

The TAP-AMP mean-approximations can be computed in $\mathrm{polylog}(n/\varepsilon)$ parallel depth. The TAP-AMP algorithms Algorithm 7 and Algorithm 5 consist of AMP iterations followed by NGD iterations. Each individual AMP or NGD iteration is computable in parallel with depth $O(\log n)$; thus, it suffices to prove $\widetilde{O}(1)$ bounds on the numbers of AMP and NGD iterations. Since this proof is rather involved and does not affect the rest of the argument, we give only a brief sketch here; the complete details appear in the full version of this paper.

▶ **Proposition 17.** *The mean-approximation algorithms Algorithm 7 and Algorithm 5 are computable with parallel depth $\mathrm{polylog}(n/\varepsilon)$, given query access to the mixed p-spin Hamiltonian and the parameter choices specified in Theorem 14 for attaining normalized Wasserstein error ε.*

4.4 Lipschitz Property of the TAP-AMP Mean Approximation for the SK Model

In this section, we prove that, for the SK model at high temperature $\beta < \beta_0 \approx 0.3753$, the TAP-AMP algorithm (Algorithm 7) for mean approximation is $O\Big(\frac{1}{\beta_0 - \beta}\Big)$-Lipschitz with respect to the tilt parameter, or external field, with high probability. We provide an analogous result for the mixed p-spin model in the full version of this paper.

Algorithm 7 Mean of the Tilted Ising Measure [17].

Input: Data $A \in \mathbb{R}^{n\times n}$, $y \in \mathbb{R}^n$, parameters $\beta, \eta > 0$, $q \in (0,1)$, iteration numbers $K_{\mathrm{AMP}}, K_{\mathrm{NGD}}$

$m_{-1} = z_0 = 0$

for $k = 0, \dots, K_{AMP} - 1$ **do**

  $m_k = \tanh(z_k), \quad b_k = \frac{\beta^2}{n}\sum_{i=1}^n (1 - \tanh^2((z_k)_i))$

  $z_{k+1} = \beta A m_k + y - b_k m_{k-1}$

$u^0 = z_{K_{\mathrm{AMP}}}$

for $k = 0, \dots, K_{NGD} - 1$ **do**

  $u_{k+1} = u_k - \eta \cdot \nabla \hat{\mathcal{F}}_{\mathrm{TAP}}(m_k^+; y, q)$

  $m_{k+1}^+ = \tanh(u_{k+1})$

return $m^+_{K_{\mathrm{NGD}}}$

Algorithm 7 consists of two phases: it first uses AMP iterations attempting to get close to the fixed point of the TAP iteration, and then it uses an NGD algorithm to close the gap to the fixed point as much as needed. Here, we will take a closer look at the two phases of Algorithm 7.

4.4.1 Phase 1: AMP phase

In the first phase, initialize $m_{-1} = 0$ and $z_0 = 0$. For $k = 0, 1, \dots, K_{\mathrm{AMP}} - 1$, set $m_k = \tanh(z_k)$ and $b_k = \frac{\beta^2}{n}\sum_{i\in[n]} \operatorname{sech}^2\big((z_k)_i\big)$, then update

$$z_{k+1} = \beta A m_k + y - b_k\, m_{k-1}. \tag{37}$$

4.4.2 Phase 2: NGD phase on TAP free energy

Initialize $u_0 := z_{K_{\mathrm{AMP}}}$. For $t = 0, 1, \dots, K_{\mathrm{NGD}} - 1$, set $m_t^+ = \tanh(u_t)$ and update

$$u_{t+1} = u_t - \eta\, \nabla_m \hat{\mathcal{F}}_{\mathrm{TAP}}(m_t^+; y, q). \tag{38}$$

The algorithm outputs $\hat{m}(A, y) := m^+_{K_{\mathrm{NGD}}} = \tanh(u_{K_{\mathrm{NGD}}})$.

▶ **Definition 18.** *For a given external field $y \in \mathbb{R}^n$ and interaction matrix $A \in \mathbb{R}^{n\times n}$, let $\hat{m}_{(K_{AMP}, K_{NGD})}(A, y)$ be the output of the algorithm after running the AMP iteration for K_{AMP} steps and then running NGD for K_{NGD} steps. We refer to $\hat{m}(y)$ as the tilted-mean computation for the external field y whenever A, K_{AMP}, and K_{NGD} are clear from the context.*

Our goal in this section is to find the temperature $\beta(A) = \beta(\|A\|_{\mathrm{op}})$ such that, for any two tilts y and $\widetilde{y}$, the TAP fixed point of the Ising model with interaction matrix A and temperature β satisfies

$$\|\hat{m}_{(K_{\mathrm{AMP}}, K_{\mathrm{NGD}})}(A, y) - \hat{m}_{(K_{\mathrm{AMP}}, K_{\mathrm{NGD}})}(A, \widetilde{y})\| \le \frac{\gamma(A)}{\beta(A) - \beta}\|y - \widetilde{y}\|.$$

▶ **Theorem 19** (Lipschitzness of AMP phase). *Assume $\gamma(A, \beta) := \beta\,\|A\|_{\mathrm{op}} + (1 + \frac{4}{3\sqrt{3}})\beta^2 < 1$. Then for all $k \ge 0$,*

$$\|\Delta z_k\| \le \frac{1}{1-\gamma}\,\|\Delta y\|, \qquad \|\Delta m_k\| \le \frac{1}{1-\gamma}\,\|\Delta y\|.$$

Lipschitzness of the full AMP+NGD algorithm for the SK model

We have proved so far that after an arbitrary number of iterations, $\|\Delta m_k\|$ is upper bounded by $\frac{1}{1-\gamma(A,\beta)}\|\Delta y\|$. Now, we prove that applying NGD after AMP preserves the same bound for the difference of the mean estimates. For convenience, and to avoid overloading the notation m_k from the AMP phase, we analyze the NGD step in the variables u_i defined by $m = \tanh(u)$ (equivalently $u = \operatorname{atanh}(m)$ component-wise).

▶ **Theorem 20** (TAP Lipschitz property)**.** *Assume* $\gamma(A,\beta) = \beta\|A\|_{\mathrm{op}} + (1+\frac{4}{3\sqrt{3}})\beta^2 < 1$ *and* $q \in (0,1)$*. Let* $0 < \eta \le 1$*. Then for every* $t \ge 0$*,*

$$\|\hat{m}(A,y) - \hat{m}(A,\widetilde{y})\| \le \frac{1}{1-\gamma(A,\beta)}\|y-\widetilde{y}\|.$$

▶ Remark 21. The key stability requirement for AMP and NGD iterations is

$$\gamma(A,\beta) = \beta\|A\|_{\mathrm{op}} + (1+\frac{4}{3\sqrt{3}})\beta^2 < 1.$$

In random matrix settings (e.g. $A \sim \mathrm{GOE}(n)$), one typically has $\|A\|_{\mathrm{op}} = 2 + o(1)$ with high probability, so any sufficiently small β ensuring $\gamma(2,\beta) < 1$ implies that the TAP-AMP mean approximation is $\frac{1}{1-\gamma(2,\beta)-c}$-Lipschitz with high probability for every sufficiently small constant c.

▶ **Corollary 22.** *Fix* $\varepsilon > 0$*. If the SK inverse temperature* β *is at most* $\beta_0 - \varepsilon$*, where* $\beta_0 \approx 0.3753$ *is the positive root of the equation*

$$\gamma(2,\beta) = 2\beta + (1+\frac{4}{3\sqrt{3}})\beta^2 = 1,$$

then the tilted-mean computation in Algorithm 7 is $O(\frac{1}{\varepsilon})$*-Lipschitz for the SK model with high probability.*

Proof. We know that $\|A\|_{\mathrm{op}}$ is concentrated around 2, and the probability that it exceeds $2+\varepsilon/\beta_0$ is $e^{-\Omega_\varepsilon(n)}$ [9]. Recall that β_0 is the positive solution of the equation $2x+(1+\frac{4}{3\sqrt{3}})x^2 = 1$. Thus, we only need to show that the parameter $\gamma = \beta\|A\|_{\mathrm{op}} + (1+\frac{4}{3\sqrt{3}})\beta^2 \le 1 - \Omega(\varepsilon)$ with high probability.

$$\begin{aligned}\beta\|A\|_{\mathrm{op}} + (1+\frac{4}{3\sqrt{3}})\beta^2 &\le \beta(2+\varepsilon/\beta_0) + (1+\frac{4}{3\sqrt{3}})\beta^2\\ &\le \varepsilon + 2\beta + (1+\frac{4}{3\sqrt{3}})\beta^2\\ &\le -\varepsilon + 2\beta_0 + (1+\frac{4}{3\sqrt{3}})\beta_0^2 = 1-\varepsilon\end{aligned}$$

Since $\mathrm{P}\{\|A\|_{\mathrm{op}} > 2+\varepsilon/\beta_0\} \le e^{-\Omega_\varepsilon(n)}$, the tilted-mean computation is $O(\frac{1}{\varepsilon})$-Lipschitz with high probability if $\beta \le \beta_0 - \varepsilon$. ◀

References

1 Arka Adhikari, Christian Brennecke, Changji Xu, and Horng-Tzer Yau. Spectral gap estimates for mixed p-spin models at high temperature. *arXiv preprint*, 2022. arXiv:2208.07844.

2 Nima Anari, Carlo Baronio, CJ Chen, Alireza Haqi, Frederic Koehler, Anqi Li, and Thuy-Duong Vuong. Parallel sampling via autospeculation. *arXiv preprint*, 2025. doi:10.48550/arXiv.2511.07869.

3 Nima Anari, Sinho Chewi, and Thuy-Duong Vuong. Fast parallel sampling under isoperimetry. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 161–185. PMLR, 2024. URL: https://proceedings.mlr.press/v247/anari24a.html.

4 Nima Anari, Ruiquan Gao, and Aviad Rubinstein. Parallel sampling via counting. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 537–548, 2024. doi:10.1145/3618260.3649744.

5 Nima Anari, Yizhi Huang, Tianyu Liu, Thuy-Duong Vuong, Brian Xu, and Katherine Yu. Parallel discrete sampling via continuous walks. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 103–116, 2023. doi:10.1145/3564246.3585207.

6 Nima Anari, Vishesh Jain, Frederic Koehler, Huy Tuan Pham, and Thuy-Duong Vuong. Entropic independence: optimal mixing of down-up random walks. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1418–1430, 2022. doi:10.1145/3519935.3520048.

7 Nima Anari, Vishesh Jain, Frederic Koehler, Huy Tuan Pham, and Thuy-Duong Vuong. Universality of spectral independence with applications to fast mixing in spin glasses. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 5029–5056. SIAM, 2024. doi:10.1137/1.9781611977912.181.

8 Nima Anari, Frederic Koehler, and Thuy-Duong Vuong. Trickle-down in localization schemes and applications. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1094–1105, 2024. doi:10.1145/3618260.3649622.

9 Greg W. Anderson, Alice Guionnet, and Ofer Zeitouni. *An Introduction to Random Matrices*, volume 118 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, 2010.

10 Michael Celentano. Sudakov–fernique post-amp, and a new proof of the local convexity of the tap free energy. *The Annals of Probability*, 52(3):923–954, 2024.

11 Xiaoyu Chen, Hongyang Liu, Yitong Yin, and Xinyuan Zhang. Efficient parallel ising samplers via localization schemes. *arXiv preprint*, 2025. doi:10.48550/arXiv.2505.05185.

12 Yuansi Chen and Ronen Eldan. Localization schemes: A framework for proving mixing bounds for markov chains. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 110–122. IEEE, 2022.

13 Laszlo Csanky. Fast parallel matrix inversion algorithms. In *16th Annual Symposium on Foundations of Computer Science (sfcs 1975)*, pages 11–12. IEEE, 1975.

14 PL Dobruschin. The description of a random field by means of conditional probabilities and conditions of its regularity. *Theory of Probability & Its Applications*, 13(2):197–224, 1968.

15 Martin Dyer, Alan Frieze, and Ravi Kannan. A random polynomial-time algorithm for approximating the volume of convex bodies. *Journal of the ACM (JACM)*, 38(1):1–17, 1991. doi:10.1145/102782.102783.

16 Ahmed El Alaoui and Andrea Montanari. An information-theoretic view of stochastic localization. *IEEE Transactions on Information Theory*, 68(11):7423–7426, 2022. doi:10.1109/TIT.2022.3180298.

17 Ahmed El Alaoui, Andrea Montanari, and Mark Sellke. Sampling from the Sherrington-Kirkpatrick Gibbs measure via algorithmic stochastic localization. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 323–334. IEEE, 2022. doi:10.1109/FOCS54457.2022.00038.

18 Ahmed El Alaoui, Andrea Montanari, and Mark Sellke. Algorithmic stochastic localization for sampling from the p-spin model. *arXiv preprint*, 2023. arXiv:2307.07130.

19 Ahmed El Alaoui, Andrea Montanari, and Mark Sellke. Sampling from mean-field gibbs measures via diffusion processes. *Probability and Mathematical Physics*, 6(3):961–1022, 2025.

20 Ronen Eldan. Thin shell implies spectral gap up to polylog via a stochastic localization scheme. *Geometric and Functional Analysis*, 23(2):532–569, 2013.

21 Ronen Eldan, Frederic Koehler, and Ofer Zeitouni. A spectral condition for spectral gap: fast mixing in high-temperature ising models. *Probability theory and related fields*, 182(3):1035–1051, 2022.

22 Jiaojiao Fan, Bo Yuan, and Yongxin Chen. Improved dimension dependence of a proximal algorithm for sampling. In *The Thirty Sixth Annual Conference on Learning Theory*, pages 1473–1521. PMLR, 2023. URL: https://proceedings.mlr.press/v195/fan23a.html.

23 Weiming Feng, Thomas P Hayes, and Yitong Yin. Distributed metropolis sampler with optimal parallelism. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2121–2140. SIAM, 2021. doi:10.1137/1.9781611976465.127.

24 Andreas Galanis, Alkis Kalavasis, and Anthimos Vardis Kandiros. On sampling from ising models with spectral constraints. *arXiv preprint*, 2024. arXiv:2407.07645, doi:10.48550/arXiv.2407.07645.

25 Andreas Galanis, Daniel Štefankovič, and Eric Vigoda. The complexity of approximating averages on bounded-degree graphs. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1345–1355. IEEE, 2020.

26 Reza Gheissari and Aukosh Jagannath. On the spectral gap of spherical spin glass dynamics. *arXiv preprint*, 2016. arXiv:1608.06609.

27 Roy J Glauber. Time-dependent statistics of the ising model. *Journal of mathematical physics*, 4(2):294–307, 1963.

28 Hengyuan Hu, Aniket Das, Dorsa Sadigh, and Nima Anari. Diffusion models are secretly exchangeable: Parallelizing ddpms via autospeculation. *arXiv preprint*, 2025. doi:10.48550/arXiv.2505.03983.

29 Brice Huang, Sidhanth Mohanty, Amit Rajaraman, and David X Wu. Weak poincaré inequalities, simulated annealing, and sampling from spherical spin glasses. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 915–923, 2025. doi:10.1145/3717823.3718236.

30 Brice Huang, Andrea Montanari, and Huy Tuan Pham. Sampling from spherical spin glasses in total variation via algorithmic stochastic localization. *arXiv preprint*, 2024. arXiv:2404.15651.

31 Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM (JACM)*, 51(4):671–697, 2004. doi:10.1145/1008731.1008738.

32 Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical computer science*, 43:169–188, 1986. doi:10.1016/0304-3975(86)90174-X.

33 Chieh-Hsin Lai, Yang Song, Dongjun Kim, Yuki Mitsufuji, and Stefano Ermon. The principles of diffusion models. *arXiv preprint*, 2025. doi:10.48550/arXiv.2510.21890.

34 Holden Lee. Parallelising glauber dynamics. *arXiv preprint*, 2023. doi:10.48550/arXiv.2307.07131.

35 Yin Tat Lee and Santosh S Vempala. Eldan's stochastic localization and the kls conjecture: Isoperimetry, concentration and mixing. *Annals of Mathematics*, 199(3):1043–1092, 2024.

36 Hongyang Liu and Yitong Yin. Simple parallel algorithms for single-site dynamics. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1431–1444, 2022. doi:10.1145/3519935.3519999.

37 Marc Mézard, Giorgio Parisi, Miguel Angel Virasoro, and David J Thouless. Spin glass theory and beyond, 1988.

38 Andrea Montanari and Subhabrata Sen. A friendly tutorial on mean-field spin glass techniques for non-physicists. *Foundations and Trends in Machine Learning*, 17(1):1–173, 2024. doi:10.1561/2200000105.

39 Ketan Mulmuley, Umesh V Vazirani, and Vijay V Vazirani. Matching is as easy as matrix inversion. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 345–354, 1987. doi:10.1145/28395.383347.

40 Dmitry Panchenko. The Sherrington-Kirkpatrick model: An overview. *Journal of Statistical Physics*, 149(2):362–383, 2012.

41 Giorgio Parisi. Correlation functions and computer simulations. *Nuclear Physics B*, 180(3):378–384, 1981.

42 Lawrence Perko. *Differential equations and dynamical systems*, volume 7. Springer Science & Business Media, 2013.

43 Holger Sambale and Arthur Sinulis. Modified log-sobolev inequalities and two-level concentration. *arXiv preprint*, 2019. `arXiv:1905.06137`.

44 Andy Shih, Suneel Belkhale, Stefano Ermon, Dorsa Sadigh, and Nima Anari. Parallel sampling of diffusion models. *Advances in Neural Information Processing Systems*, 36:4263–4276, 2023.

45 Allan Sly and Nike Sun. The computational hardness of counting in two-spin models on d-regular graphs. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 361–369. IEEE, 2012. `doi:10.1109/FOCS.2012.56`.

46 Michel Talagrand. *Mean field models for spin glasses: Volume I: Basic examples*, volume 54. Springer Science & Business Media, 2010.

47 David J Thouless, Philip W Anderson, and Robert G Palmer. Solution of'solvable model of a spin glass'. *Philosophical Magazine*, 35(3):593–601, 1977.

Quantum Algorithms for Path and Cycle Containment Problems

Arjan Cornelissen
Simons Institute, University of California Berkeley, CA, USA

Amin Shiraz Gilani
QuICS, University of Maryland, College Park, MD, USA

Subhasree Patro
Eindhoven University of Technology, The Netherlands

Abstract

The quantum query complexity of subgraph-containment problems, which ask whether a given subgraph H is present in an input graph G, has been the subject of considerable study. This interest stems not only from the natural and well-motivated formulation of these problems, but also from a flurry of novel quantum algorithmic techniques that were developed specifically to solve them. Notably, even for relatively simple subgraphs, such as paths and cycles, a complete understanding of their query complexities remains elusive.

In this work, we consider several variants of path- and cycle-containment problems in the adjacency matrix model, where we search for paths or cycles of constant length $k \in O(1)$. We compare the settings where the graphs are directed or undirected, where the goal is to detect or find the existence of a path/cycle, and where the path/cycle we're looking for has length exactly k, or at most k. We also consider several promise versions of these problems, where we know beforehand that the input graph has a certain structure. We characterize the relative difficulty of these variants of the path- and cycle-containment problems, by relating them to one another using randomized reductions, and grouping them into several equivalence classes.

When we restrict our attention to path-containment problems, this implies a dichotomy result. Some of the path-containment problems can be solved using a linear number of queries, and all the others are equivalent to one another (and additionally to several cycle-containment problems as well) under randomized reductions and up to constant multiplicative overhead. For the latter equivalence class, we prove a novel quantum-walk-based algorithm that achieves query complexity $\widetilde{O}(n^{3/2-\alpha_k})$, where $\alpha_k \in \Theta(c^{-k})$ and $c = \sqrt{3+\sqrt{17}}/2 \approx 1.33$, beating the previous best upper bound $O(n^{3/2})$ on its query complexity. We also provide a conditional lower bound based on the graph-collision problem, which implies that this equivalence class does not admit linear-query quantum algorithms unless graph collision admits an $O(\sqrt{n})$ query algorithm.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases Quantum algorithms, query complexity, graph problems, fine-grained reductions

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.72

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2605.09017`

Funding *Arjan Cornelissen*: Supported by a Simons-CIQC postdoctoral fellowship through NSF QLCI Grant No. 2016245.
Amin Shiraz Gilani: Acknowledges support from the U.S. Department of Energy, Office of Science, Accelerated Research in Quantum Computing, Fundamental Algorithmic Research toward Quantum Utility (FAR-Qu) and the National Institute of Standards and Technology (NIST).
Subhasree Patro: Acknowledges the support from the Dutch Ministry of Education, Culture, and Science through Gravitation project "Challenges in Cyber Security - 024.006.037" for this work.

Acknowledgements We would like to thank Andrew Childs and Matthew Coudron for many useful discussions.



© Arjan Cornelissen, Amin Shiraz Gilani, and Subhasree Patro;
licensed under Creative Commons License CC-BY 4.0

1 Introduction

The subgraph containment problem asks whether a given (undirected) subgraph H is contained in an (undirected) input graph G. In this work, we consider subgraphs H of constant size that are known ahead of time, and input graphs G consisting of n vertices that we can access through adjacency-matrix queries. Now, we wish to decide whether the subgraph H is present in the input graph G, while making a minimal number of queries to the adjacency matrix of G. We denote this problem concisely by $\texttt{Subgraph}^H$, and the minimal number of queries to the input required to solve it is known as its query complexity. We also consider a restricted version of this problem where we select several vertices $v_1, \ldots, v_m$ from H and additionally require that these vertices get mapped to predetermined positions $s_1, \ldots, s_m$ in G. Under this restriction, we then ask whether H is present in G, and we denote the resulting problem by $\texttt{Subgraph}^H_{v_1,\ldots,v_m}$.

In the randomized setting, we decide which edges to query by a (classical) randomized algorithm, and we require the resulting algorithm to output the correct answer with probability at least $2/3$. The randomized query complexity of the unrestricted subgraph containment problem, denoted by $\mathsf{R}(\texttt{Subgraph}^H)$, satisfies $\Theta(n^2)$ for any non-empty graph H. Moreover, it is not hard to show that the restricted problems are classified into three distinct classes, where the randomized query complexities $\mathsf{R}(\texttt{Subgraph}^H_{v_1,\ldots,v_m})$ are $\Theta(1)$, $\Theta(n)$ and $\Theta(n^2)$, respectively.

In contrast, in the quantum setting, we can query the adjacency matrix coherently in superposition, and we aim to give a quantum algorithm whose final measurement outputs the right answer with probability at least $2/3$. The corresponding quantum query complexities, denoted by $\mathsf{Q}(\texttt{Subgraph}^H)$ for the unrestricted problem, and $\mathsf{Q}(\texttt{Subgraph}^H_{v_1,\ldots,v_m})$ for the restricted problem, seem to have a much richer structure than their classical counterparts. This is the focus of this work.

The study of the constant-size subgraph-containment problem in the quantum setting was pioneered by Magniez, Szegedy and Santha [24], who built on a quantum walk algorithm by Ambainis [2] to show that the existence of any subgraph H consisting of k vertices can be detected using $\widetilde{O}(n^{2-2/k})$ queries. This result was subsequently independently improved by Zhu [28], and Lee, Magniez and Santha [21], resulting in the current state-of-the-art complexity for general subgraphs H through the following learning-graph-based result.

▶ **Theorem 1** ([21, Theorem 4.4]). *Let H be an undirected graph with $k \geq 3$ vertices, m edges, and minimum vertex-degree $d \geq 1$, where $k, m, d \in O(1)$. Then, $\mathsf{Q}(\texttt{Subgraph}^H) \in O\left(n^{2-2/k-t}\right)$, where*

$$t = \max\left\{\frac{k^2 - 2(m+1)}{k(k+1)(m+1)}, \frac{2k-d-3}{k(d+1)(m-d+2)}\right\}.$$

Even though this result gives a non-trivial upper bound on the query complexity for the subgraph containment problem for any subgraph H, it is not expected that this bound is optimal in general. Indeed, for several specific families of subgraphs, we know better bounds than those provided by Lee, Magniez and Santha. Examples of such subgraph families include the line graphs with k edges (L_k) and cycle graphs with k edges (C_k), for which we refer to the subgraph-containment problems as $\texttt{Path}^{=k} := \texttt{Subgraph}^{L_k}$ and $\texttt{Cycle}^{=k} := \texttt{Subgraph}^{C_k}$, respectively.

The path-containment problem was pioneered by Childs and Kothari [14], who showed that it can be solved using $\widetilde{O}(n^{3/2-1/\lceil k/2\rceil})$ quantum queries. This was later improved by Belovs and Reichardt [6] to $\mathsf{Q}(\texttt{Path}^{=k}) \in \Theta(n)$, for all $k \in O(1)$. Their algorithm involves

reducing this problem to the promise problem, which we denote by $\texttt{PromPath}^{\leq k}_{s,t}$, where two vertices s and t are fixed, and the task is to distinguish between the existence of a path between s and t of length at most k, and no path between s and t at all. They used a span-program approach to show that $\mathsf{Q}(\texttt{PromPath}^{\leq k}_{s,t}) \in O(n\sqrt{k})$, even for non-constant k.

Despite Belovs's and Reichardt's remarkable progress on the path-containment problem, subsequent efforts to generalize their techniques were unsuccessful. Specifically, all attempts to solve the path-containment problem in the directed setting, or a version of their promise problem where one is required to not just detect the existence of a path, but actually output it, have not resulted in linear-query algorithms. In this work, we argue why these efforts are unlikely to work, with the following reasoning containing two parts. First, we show that all these directed and finding versions of the path-containment problem are equivalent up to randomized reductions (Section 3), and then we show a conditional quantum query lower bound for all these problems based on the graph-collision problem (Section 6). Since it is considered unlikely that we can make progress on the graph-collision problem with the current techniques, we conclude it's unlikely that we can make progress on these generalizations of Belovs's and Reichardt's work with the current techniques as well.

On the topic of cycle-containment problems, the problem of detecting 3-cycles (more commonly known as the triangle-detection problem) has played a central role in the literature on quantum query complexity. The study of this problem gave rise to many novel quantum algorithmic frameworks, and has been improved multiple times since it was first considered in [9]. Currently, the best-known algorithm is by Carette, Laurière and Magniez [11], who removed the log-factors from the algorithm by Le Gall [20].

▶ **Theorem 2** ([20, 11]). $\mathsf{Q}(\texttt{Cycle}^{=3}) \in O(n^{5/4})$.

The more general constant-length cycle-finding problem was also considered by Childs and Kothari [14], who showed the following improved upper bounds for even values of k.

▶ **Theorem 3** ([14, Theorem 4.11,4.12]). *Let $4 \leq k \in \mathbb{N}$ be even. Then,* $\mathsf{Q}(\texttt{Cycle}^{=k}) \in \widetilde{O}\left(n^{\frac{3}{2}-\frac{k-2}{k(k+2)}}\right)$. *Moreover,* $\mathsf{Q}(\texttt{Cycle}^{=4}) \in \widetilde{O}(n^{5/4})$.

Finally, Cade, Montanaro and Belovs [10] considered a promise version of the cycle-containment problem, where they fix a vertex s in the input graph, and then distinguish between graphs with a cycle of length at most k passing through s and graphs with no cycles at all. Much like the promise path problem, they present an algorithm making $O(n\sqrt{k})$ queries, even for non-constant values of k.

As in the path-containment setting, all attempts to generalize the above results to the directed setting proved to be futile. Moreover, generalizing the promise version of the problem considered by Cade, Montanaro and Belovs to the finding setting, where one additionally has to output the cycle, proved elusive as well. In this work, we connect the cycle-containment problems rigorously to the path-containment problems through randomized reductions, explaining in a similar fashion why these generalization efforts are unlikely to succeed.

From the lower-bound perspective, progress is hampered by the *certificate barrier* [27, 26], which states that lower bounds obtained through the non-negative-weighted adversary bound are upper bounded by the geometric average of the worst-case positive and negative certificate sizes. For all subgraph containment problems where the subgraph H is of constant size, the positive certificate sizes are of constant size as well, and therefore the best possible lower bound on the quantum query complexity we can obtain through the non-negative-weighted adversary bound is, up to constants, equal to $\sqrt{n^2 \cdot 1} = n$.

Even though there are other methods available for proving lower bounds on quantum query complexity, like the polynomial method [4], and the general adversary bound [17], applying these techniques to the subgraph containment problem has so far not been fruitful. As such,

proving any unconditional super-linear lower bound on the quantum query complexity for any subgraph containment problem is a major open problem in this area that has been open for about two decades.

Despite (or perhaps because of) these barriers, alternative progress was made by Balodis and Iraids [3] in showing a conditional lower bound for the triangle-detection problem, based on the hardness of the graph-collision problem. This problem is parametrized by a graph G on n vertices, which is completely known ahead of time. The input to the problem is a bit sting x that we have query access to, where each bit of x is associated with a distinct vertex of G. The graph collision problem on G asks whether there exists an edge in G for which both endpoints are labeled by a 1 in x. We refer to this problem as $\mathtt{GC}_G$, and we write $\mathsf{Q}(\mathtt{GC}_n)$ for the maximum of $\mathsf{Q}(\mathtt{GC}_G)$, over all graphs G on n vertices.

The best generic algorithm for this problem was given by Magniez, Szegedy and Santha [24, Theorem 3], which is essentially based on the algorithm of Ambainis [2] and has query complexity $\mathsf{Q}(\mathtt{GC}_n) \in O(n^{2/3})$. This complexity is believed by some in the community to be optimal. However, the best-known lower bound is only $\mathsf{Q}(\mathtt{GC}_n) \in \Omega(\sqrt{n})$, where further progress is again impeded by the certificate barrier.

Balodis and Iraids embedded the OR of n instances of the graph-collision problem into a single instance of the triangle-detection problem on $3n$ vertices. In doing so, they prove a conditional lower bound on the family of cycle-containment problems.

▶ **Theorem 4** ([3, Theorem 1]). $\mathsf{Q}(\mathtt{Cycle}^{=3}) \in \Omega(\sqrt{n} \cdot \mathsf{Q}(\mathtt{GC}_n))$.

This result implies a plausible path toward lower bounding the query complexity of the triangle-detection problem. Indeed, any lower bound $\omega(\sqrt{n})$ for the graph-collision problem would immediately imply a super-linear lower bound $\omega(n)$ for the triangle-detection problem.

In this work, we provide similar graph-collision-based conditional lower bounds for the families of path- and cycle-containment problems. This solidifies the role of the graph-collision problem in the landscape of subgraph-containment problems, as a non-trivial lower bound for graph-collision would imply non-trivial lower bounds for both cycle- and path-containment problems.

1.1 Contributions

In this work, we provide a rigorous study of the different variants of subgraph-containment problems, with a specific focus on path- and cycle-containment problems. We present a number of simpler *observations* based on prior work that help paint the landscape of these problems, and we obtain three *main results* that improve our understanding of their query complexities.

For the general constant-size subgraph-containment problem, we consider relations between the undirected and the directed setting, as well as between their detection and finding versions. Even though it is not explicitly analyzed by Lee, Magniez and Santha, we make the simple observation that their learning-graph-based approach readily generalizes to the directed setting.

▶ **Observation 5** (Informal version of Proposition 25). *Theorem 1 holds too whenever the subgraph H and the input graph are directed.*

Next, we also observe that for non-promise subgraph-containment problems, detection and finding are equivalent up to randomized reductions and constant multiplicative overhead.

▶ **Observation 6** (Informal version of Lemma 23). *For any (directed/undirected) subgraph H, the quantum query complexities of the finding and detection versions of the non-promise (possibly restricted) subgraph-containment problem with subgraph H are equal up to a multiplicative constant.*

We note that the above only holds for non-promise problems, i.e., where all graphs are valid inputs to the problem. That is, one can only have separations between the detection and finding versions of a subgraph-containment problem in a setting where the input satisfied some promise known a priori. Indeed, in such promise settings, there are several graph problems where the finding and detection versions seem to be very much distinct. For instance, there is a quantum algorithm that makes polylogarithmic queries to output the (first bit of the) EXIT vertex in the Welded tree graph [12], while outputting an ENTRANCE-to-EXIT path in this graph takes polynomially many queries for a certain natural class of quantum algorithms [13]. Moreover, as we will see, the promise problem $\texttt{PromFindPath}^{=k}_{s,t}$ would be polynomially harder than $\texttt{PromPath}^{=k}_{s,t}$ assuming graph collision is harder than its trivial lower bound.

We also show that there is a generic randomized reduction from the undirected version of a subgraph-containment problem to the directed one.

▶ **Observation 7** (Informal version of Lemma 24). *Every directed subgraph-containment problem is at least as hard (in terms of randomized/quantum query complexity) as its undirected counterpart.*

Next, we focus specifically on the path- and cycle-containment problems. For these problems, we consider the restricted and unrestricted versions of the problem, where in the restricted versions, we fix the start and end vertices s and t for the path-containment problem, and a single vertex s on the cycle for the cycle-finding problem. We also consider the directed versions of these problems, and the finding versions where we ask to output the path or cycle. Finally, we consider versions where we search for a path or cycle of length exactly k, or at most k. We denote the directed and finding versions with the keywords `Dir` and `Find`, we attach a superscript $= k$ or $\leq k$ to indicate the length constraint, and we supply a subscript s or s, t for the restricted versions. Thus, for instance, the problem $\texttt{FindDirPath}^{=k}_{s,t}$ is the finding version of the directed path-finding problem between s and t, of length exactly k.

Akin to Belovs and Reichardt [6], and Cade et al. [10], we also consider promise versions of this problem. In the restricted path problem, i.e., where we fix the start and end vertex s and t, the promise version satisfies that a path of the required length exists between s and t, or s and t are not connected at all. Similarly, in the restricted cycle-finding problem through a vertex s, either a cycle through s exists with the required length, or no cycle through s exists at all. We note that this promise is weaker than the promise considered in Cade et al. [10], and so our algorithmic results for this problem are stronger. Finally, for the unrestricted cycle-finding problem, we also consider a promise version, where either a cycle exists of the required length, or there exists no cycle at all (i.e., the graph is a forest). These problems are indicated with the `Prom` keyword, i.e., $\texttt{PromDirCycle}^{\leq k}_{s}$ is the promise problem of detecting a directed cycle through s of length at most k.[1]

Our first observation in this direction is that every `PromFind`-version of these problems requires the same number of queries as the corresponding `Find`-version.

[1] Note that we don't consider a promise version of the unrestricted path-containment problem, because we don't see a meaningful way of defining it.

▶ **Observation 8** (Informal version of Lemmas 28 and 38)**.** *Let* P *be a (restricted/unrestricted, directed/undirected) path- or cycle-containment problem, with length constraint* $= k$ *or* $\leq k$ *with* $k \in O(1)$*. Then, the finding version of* P *is equivalent to the promise-finding version of* P*, up to randomized reductions and multiplicative constants.*

Observations 6 and 8 show that for all path- and cycle-containment problems P, we have that P, FindP and PromFindP are equivalent up to randomized reductions and multiplicative constants. As such, these three problems fall in the same equivalence class, and it suffices to restrict our attention to P and PromP in what follows.

Now, we arrive at the first main result of this work. We classify all remaining path- and cycle-containment problems into several equivalence classes via randomized reductions between them. We aid the reader by providing a graphical overview of these reductions in Figure 1, and we obtain the following classification.

▶ **Main result 9.** *The unrestricted and undirected path-containment problem, and the undirected promise versions of the path- and cycle-containment problems admit linear-query quantum algorithms. All other path-containment problems are equivalent, and all other cycle-containment problems can be classified into at most 5 equivalence classes. See Figure 1.*

For some reductions we use the color-coding technique, introduced by Alon, Yuster and Zwick [1]. The idea is to assign distinct colors to the vertices of the path/cycle graph L_k/C_k, and then to randomly assign these colors to the vertices of the input graph G. Next, we only keep an edge in G if the colors of the two vertices it connects are associated to two vertices in the path/cycle graph that are connected by an edge too. The resulting color-coded graph becomes a layered graph in the path-containment case, and a layered cycle graph in the cycle-containment case, as displayed in Figure 2. With at least constant probability, a path/cycle in the original graph gets mapped to a path/cycle in the layered graph that straddles across all the layers.

To show relations between the various subgraph-containment problems, we cleverly make subtle modifications to the input graphs, oftentimes combined with the aforementioned color-coding technique. For instance, we can insert layers into a layered graph to show that one can find short paths using an algorithm that finds longer paths. Similarly, we can merge the vertices s and t into a single vertex s to convert an st-path containment problem into a cycle-containment problem through the vertex s, and similarly we can separate a vertex s into two vertices s and t to prove a reduction in the reverse direction.

All these conversions, however, are subject to subtle caveats. For instance, in a layered cycle graph with k layers, a cycle of length k is guaranteed to go through all layers whenever k is odd, whereas when k is even it can exist between two consecutive layers. Similarly, when we merge two vertices s and t into a single vertex s, a cycle through s in the resulting graph does not necessarily imply that the original graph contains an st-path. We obtain our classification, i.e., Main result 9, by carefully checking which reductions can be made to work despite these caveats. Since we don't expect Figure 1 to collapse to a single island, we don't expect these subtleties to be artifacts of the techniques used, but rather highlight the fundamental relations between these problems.

The overview of reductions presented in Figure 1 reveals some interesting relations. For instance, it shows that one can reduce the $\texttt{DirCycle}_s^{\leq k}$-problem to the $\texttt{Cycle}_s^{\leq k}$-problem, through a series of reductions that involve the $\texttt{DirCycle}_s^{=k}$- and $\texttt{Cycle}_s^{=k}$-problems. We remark here that making this connection without going through the intermediate steps is not immediate, since it's not clear how one can remove directions in a layered graph while preventing the introduction of new cycles. This highlights the benefit of this classification effort, as it uncovers relations between problems that would otherwise not be easily obtained.

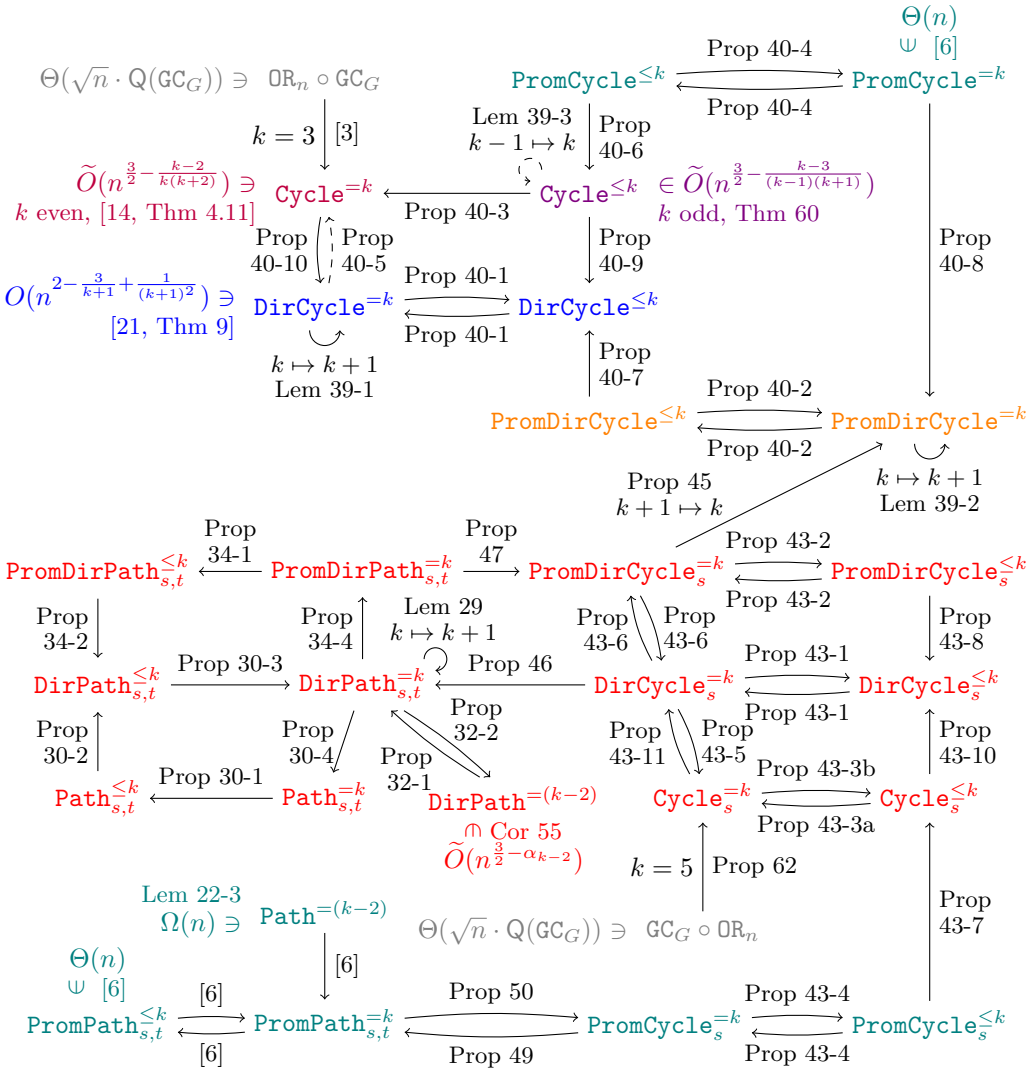


Figure 1 Randomized reductions between problems for $k \geq 3$. The dashed connections hold only for odd values of k. For the graph-collision problems, we can take any graph G with n vertices. All problems' randomized query complexities are $\Theta(n^2)$. All complexity bounds stated in the figure are upper and lower bounds on the quantum query complexity. In the upper bound for $\mathsf{Q}(\mathtt{DirPath}^{=(k-2)})$, $\alpha_k \in \Theta(c^{-k})$, with $c \approx 1.33$. If two problems have the same color, they have the same quantum query complexity up to constants.

Next, the classification of path- and cycle-finding problems displayed in Figure 1 motivates the search for new quantum algorithms and query lower bounds for problems in these equivalence classes specifically. For instance, the $\mathtt{DirPath}^{=k}$-problem is part of an equivalence class containing 12 other path- and cycle-containment problems, hence an improved algorithm for this problem immediately provides an algorithm for all the others as well. Moreover, this equivalence class includes all directed versions of the path problems, as well as the finding versions of the promise-path problems, i.e., exactly the problems for which generalizing Belovs's and Reichardt's span-program-based construction failed.

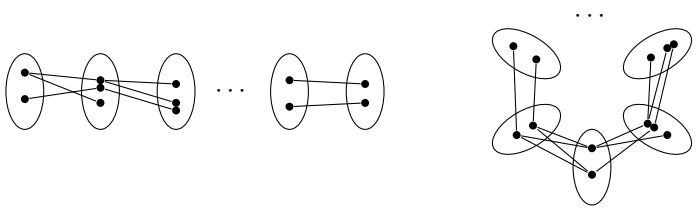


Figure 2 A layered graph (left) and a layered cycle graph (right). These can be obtained from general graphs using the color-coding technique, where a path of length k (resp. cycle of length k) is preserved with constant probability.

Our main algorithmic result is a novel quantum-walk-based algorithm for this $\texttt{DirPath}^{=k}$-problem, and hence for all problems in the equivalence class highlighted in red in Figure 1. The best previously-known algorithm for any of the problems in this equivalence class is an algorithm that finds the shortest path between s and t, if it exists, and thereby solves the $\texttt{Path}_{s,t}^{\leq k}$-problem. This algorithm was originally developed by Dürr et al. [16], making $\widetilde{O}(n^{3/2})$ queries, and the log-factors were subsequently removed by a series of works culminating in a paper by Lin and Lin [22, Theorem 13]. We note that Beigi and Taghavi also developed $O(n^{3/2})$-query algorithms for the $\texttt{Cycle}_s^{=k}$ and $\texttt{Cycle}_s^{\leq k}$-problems [5, Proposition 9.iv-v]. Our result provides an improved algorithm for all these problems.

▶ **Main result 10** (Corollary 55). *Let $5 \leq k \in \mathbb{N}$. Then, $\mathsf{Q}(\texttt{DirPath}^{=k}) \in \widetilde{O}(n^{3/2-\alpha_k})$, where $\alpha_k \in \Theta(c^{-k})$ as $k \to \infty$, with $c = \sqrt{3+\sqrt{17}}/2 \approx 1.33$.*

The main algorithmic idea is to reduce the directed-path-detection problem of length k to the same problem with length $k-2$. To that end, we employ the MNRS-framework [23] to search for vertices that have at least one incoming edge, and those that have at least one outgoing edge. Then, it remains to find a pair of such vertices that has a path of length $k-2$ between them, which we then solve recursively. This results in a nested quantum-walk construction, where we optimize the parameters of the quantum-walk framework to minimize the resulting query complexity. The resulting minimization problem involves solving a recurrence relation, whose solution yields an exponent that satisfies $3/2 - \Theta(c^{-k})$ for large values of k.

We additionally provide an improved algorithm for the $\texttt{Cycle}^{\leq k}$-problem, which is also displayed in Figure 1. A slight modification of Childs and Kothari [14] yields the following result.

▶ **Observation 11** (Theorem 60). *Let $5 \leq k \in \mathbb{N}$ be an odd integer. Then, $\mathsf{Q}(\texttt{Cycle}^{\leq k}) \in \widetilde{O}(n^{\frac{3}{2}-\frac{k-5}{(k-1)(k+3)}})$.*

Finally, we focus on the lower bounds. Our main result in this direction is relating the cycle-finding problem through a particular vertex s to the graph-collision problem. In particular, through the reductions highlighted in Figure 1, we emphasize that this result also provides a conditional lower bound on the $\texttt{DirPath}^{=k}$-problem, and consequently on all path-containment problems that are in the same equivalence class.

▶ **Main result 12** (Proposition 62). *Let $5 \leq k \in \mathbb{N}$ be an odd integer. Then, $\mathsf{Q}(\texttt{Cycle}_s^{\leq k}) \in \Omega(\sqrt{n} \cdot \mathsf{Q}(\texttt{GC}_n))$.*

Even though the resulting lower bound is identical to the lower bound derived by Balodis and Iraids for the triangle-detection problem, the underlying construction is profoundly different. Whereas Balodis and Iraids embed n instances of the graph-collision problem into a

single instance of triangle-detection, we embed a single instance of the graph-collision problem composed with the OR-function into a single instance of the 5-cycle-detection problem through a fixed vertex s. In other words, Balodis and Iraids prove $\mathsf{Q}(\texttt{Cycle}^{=3}) \in \Omega(\mathsf{Q}(\texttt{OR}_n \circ \texttt{GC}_n))$, whereas we prove that $\mathsf{Q}(\texttt{Cycle}_s^{=5}) \in \Omega(\mathsf{Q}(\texttt{GC}_n \circ \texttt{OR}_n))$, where $\texttt{OR}_n$ is the boolean functions encoding the OR-function on n bits.

Finally, we remark that combining all our results provides conditional separations between the $\texttt{Path}^{=k}$ and $\texttt{DirPath}^{=k}$ problems, the $\texttt{PromDirPath}_{s,t}^{=k}$ and $\texttt{PromPath}_{s,t}^{=k}$ problems, and the $\texttt{PromFindPath}_{s,t}^{=k}$ and $\texttt{PromPath}_{s,t}^{=k}$ problems. Indeed, any non-trivial lower bound on the graph-collision problem implies a query separation between each pair of these problems. This in particular suggests that Belovs's and Reichardt's span-program-based approach can likely not be generalized to either the directed or finding settings, and, as such, that the undirected promise problems are indeed likely to be much easier than their directed and non-promise counterparts.

2 Preliminaries

2.1 Notation

Let $\mathbb{N} = \{1, 2, \ldots\}$ be the set of all positive integers, and $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$. For any integer $n \in \mathbb{N}$, we define $[n] \coloneqq \{1, \ldots, n\}$ and $[n]_0 \coloneqq \{0, 1, \ldots, n\}$.

Let $g : \mathbb{R} \supseteq D_g \to \mathbb{R}_{\geq 0}$. We define $O(g)$ to be the set of functions $f : D_g \supseteq D_f \to \mathbb{C}$ such that there exist $C, M > 0$ such that for all $x \in D_f$, $x > M \Rightarrow |f(x)| \leq C \cdot g(x)$. Similarly, we define $\widetilde{O}(g)$ to be the set of functions $f : D_g \supseteq D_f \to \mathbb{C}$ such that there exist $C, M, k > 0$ such that for all $x \in D_f$, $x > M \Rightarrow |f(x)| \leq C \cdot g(x) \cdot \log^k(g(x))$. We define $\Omega(g)$ and $\widetilde{\Omega}(g)$ analogously, but with the conditions being $|f(x)| \geq C \cdot g(x)$ and $|f(x)| \geq C \cdot g(x)/\log^k(g(x))$, respectively. Finally, we write $\Theta(g) := O(g) \cap \Omega(g)$, and $\widetilde{\Theta}(g) := \widetilde{O}(g) \cap \widetilde{\Omega}(g)$.

We denote undirected graphs by $G = (V, E)$, where every edge $\{v, w\} = e \in E$ is a set of two vertices, with $v, w \in V$. Similarly, we denote directed graphs by $G = (V, A)$, where every arc $(v, w) = a \in A$ is an ordered pair of vertices, with $v, w \in V$. If $V' \subseteq V$, then $G - V'$ denotes the graph G with all vertices from V' and their adjacent edges removed. All (undirected/directed) graphs considered in this work are simple, i.e., there are no duplicate edges and arcs. However, in the directed setting, we do allow for both arcs in opposite directions to exist simultaneously between two vertices.

We consider the adjacency-matrix model, where any algorithm can make queries to the entries of the adjacency matrix of an input graph G. Equivalently, for an undirected graph, the algorithm can supply a set of two vertices to an oracle, that outputs whether an edge exists between the two, and similarly in the directed setting the algorithm can ask whether an arc exists between two vertices from an ordered pair. In the randomized setting, the algorithm can employ randomness to select which vertices to query. In the quantum setting, the vertex pairs can be queried coherently in superposition, e.g., for an undirected graph $G = (V, E)$, the oracle behaves as a unitary O_G that acts as

$$O_G : |v, w\rangle\,|b\rangle \mapsto |v, w\rangle\,|b \oplus [\{v, w\} \in E]\rangle\,, \qquad \forall v, w \in V, b \in \{0, 1\}.$$

We refer the reader to a quantum computing textbook for more details on the quantum computational model, e.g., [25]. The randomized and quantum query complexities of a given problem are the minimal numbers of queries required to solve the problem in these respective computational models.

Finally, we use regular expressions to denote sets of problems. That end, for two expressions $\texttt{A}$ and $\texttt{B}$, we write $(\texttt{A}|\texttt{B})$ to denote the set of expressions $\{\texttt{A}, \texttt{B}\}$. We use this in combination with concatenation, i.e., $\texttt{A}(\texttt{B}|\texttt{C})$ denotes the set $\{\texttt{AB}, \texttt{AC}\}$. In particular, some of the expressions can be empty, i.e., $(|\texttt{A})\texttt{B}$ denotes the set $\{\texttt{B}, \texttt{AB}\}$.

2.2 Subroutines

We begin with a quantum subroutine that approximates the hamming weight of a Boolean string with a an error parameter ϵ.

▶ **Theorem 13** (Approximate quantum counting [8, Theorem 15]). *Let $f : \{0,1\}^n \to \{0,1\}$ and $\epsilon \in (0,1]$. Let $t = |f^{-1}(1)|$. Then, there is a quantum algorithm that outputs $\tilde{t}$ such that $|t - \tilde{t}| \leq \epsilon t$ with probability at least $1 - \delta$ using $O\left(\frac{1}{\epsilon}\sqrt{n/t}\log\frac{1}{\delta}\right)$ queries to f.*

We will mainly need to use the approximate counting algorithm for approximating the number of edges in a graph and approximating the degree of a vertex. For that reason, we will need the following corollaries.

▶ **Corollary 14** ([14, Corollary 4.1]). *Let $A : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ be an adjacency matrix representation of an undirected graph G, $\epsilon \in (0,1]$ be a constant and $\overline{m}$ be an integer. Then, there is a quantum algorithm ApproxEdgeCount that accepts G if the number of edges $|E(G)| = |A|$ in G are at least $(1+\epsilon)\overline{m}$ and rejects G if $|E(G)|$ is at most $(1-\epsilon)\overline{m}$ with probability at least $1 - \delta$ using $O\left(\frac{1}{\epsilon}\sqrt{n^2/\overline{m}}\log\frac{1}{\delta}\right)$ queries to A.*

▶ **Corollary 15.** *Let $A : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ be an adjacency matrix representation of an undirected graph G, v be a vertex in G, $\epsilon \in (0,1]$ be a constant and $\overline{d}$ be an integer. Then, there is a quantum algorithm ApproxDegreeCount that accepts (G, v) if the degree $\deg(v)$ of v in G are at least $(1+\epsilon)\overline{d}$ and rejects G if $\deg(v)$ is at most $(1-\epsilon)\overline{d}$ with probability at least $1 - \delta$ using $O\left(\frac{1}{\epsilon}\sqrt{n/\overline{d}}\log\frac{1}{\delta}\right)$ queries to A.*

We will also need a subroutine that retrieves all the marked elements of a function.

▶ **Lemma 16** ([14, Lemma 4.1]). *Let $f : [n] \mapsto \{0,1\}$ and $t = |f^{-1}(1)|$. Then, there is a quantum algorithm that computes $f^{-1}(1)$ using $O(\sqrt{n(t+1)})$ queries to f.*

The following lemma relates the query complexity of problems that can be related by a randomized reduction with one-sided error.

▶ **Lemma 17.** *Let P and P' be arbitrary decision problems. Let G be an instance of P and let G' be obtained from G via an efficient randomized reduction, i.e., each query to G' can be made using $O(1)$ queries to G. Suppose that if G is a YES instance of P, then G' is a YES instance of P' with probability at least $p \in \Omega(1)$, and if G is a NO instance of P, then G' is a NO instance of P' with probability 1. Then, for $\mathsf{M} \in \{\mathsf{R}, \mathsf{Q}\}$, we have $\mathsf{M}(\mathsf{P}) \in O(\mathsf{M}(\mathsf{P}'))$.*

Proof. Let $\mathcal{A}$ be the optimal (randomized/quantum) algorithm for the problem P'. We first construct a new algorithm $\mathcal{B}$ with majority voting, where we run $\mathcal{A}$ on the same input a total of $\Theta(\log(1/p))$ times, so that it decides between YES and NO inputs with probability at least $1 - p/4$.

Now, we run the algorithm $\mathcal{B}$ on an instance G' generated through the randomized reduction. Then, if G is a positive input, this procedure will succeed with probability at least $p(1 - p/4) > 3p/4$, whereas if G is a negative input, this procedure will succeed with probability at most $p/4$. Thus, by running this procedure a total of $\Theta(1/p)$ times, we can distinguish between these two cases with high probability. As such, the total number of calls to $\mathcal{A}$ is $\Theta((1/p)\log(1/p))$, which is constant as $p \in \Omega(1)$. Thus, $\mathsf{M}(\mathsf{P}) \in O(\mathsf{M}(\mathsf{P}'))$, for $\mathsf{M} \in \{\mathsf{Q}, \mathsf{R}\}$. ◀

▶ Remark 18. Note that, to be able to argue sub-quadratic quantum query bounds for graph problems on n vertices through randomized reductions, these reductions must be query-efficient. In the reductions we use throughout this work, we express the edge set of the new graph in terms of the edge set of the original graph, such that checking a specific edge in the new graph requires $O(1)$ queries to the edge set of the original graph, unless explicitly stated otherwise.

2.3 Quantum walks

Throughout this paper, we will be making use of the MNRS-framework for quantum walks, developed by Magniez, Nayak, Rolland and Santha [23]. We recall their result here.

▶ **Theorem 19** ([23, Theorem 3]). *Let P be a reversible ergodic Markov chain over a state space V, with spectral gap δ. Let $M \subseteq V$ be a marked subset, such that if $M \neq \varnothing$, then $\mathbb{P}_{v\sim\pi}[v \in M] \geq \varepsilon > 0$, where π is the stationary distribution of P. Let S be such that the number of queries required to prepare a q-sample of $|\pi\rangle$ up to norm-error at most δ is $\widetilde{O}(\mathsf{S}\text{polylog}(1/\delta))$. Similarly, let U be such that the number of queries required to prepare a q-sample over the neighbors of a given vertex $v \in V$ with norm-error at most δ is $\widetilde{O}(\mathsf{U}\text{polylog}(1/\delta))$. Finally, let C be the number of queries required to check whether a given state $v \in V$ is marked, with high probability. Then, we can decide whether M is empty or non-empty with a total number of queries that satisfies*

$$\widetilde{O}\left(\mathsf{S} + \frac{1}{\sqrt{\varepsilon}}\left(\frac{1}{\sqrt{\delta}}\mathsf{U} + \mathsf{C}\right)\right).$$

On several occasions, we will recursively use this construction, which is known as a nested quantum walk. This has been considered ample times in previous works, for instance [14, 19]. These constructions typically suffer from polylogarithmic overhead, but recent work has shown that this can be avoided in certain settings [18, 15]. We leave using these more recent techniques to remove the polylogarithmic overhead from our algorithms for future work.

3 Reductions between subgraph containment problems

We start by formally introducing the different versions of the subgraph-containment problems we consider in this work.

▶ **Definition 20** (Subgraph containment). *We define several computational problems.*

1. *Let $H = (V_H, E_H)$ be a known undirected graph, and suppose we have access to an undirected graph $G = (V, E)$ through adjacency-matrix queries. Let $m \in \mathbb{N}_0$, $v_1, \ldots, v_m \in V_H$ and $s_1, \ldots, s_m \in V$ all distinct. The subgraph containment problem asks whether there exists an injection $\varphi : V_H \to V$, such that $\varphi : v_j \mapsto s_j$, for all $j \in [m]$, and such that the induced map $\varphi' : \binom{V_H}{2} \to \binom{V}{2}$ acting as $\{v, w\} \mapsto \{\varphi(v), \varphi(w)\}$ satisfies $\varphi'(E_H) \subseteq E$. The resulting boolean function, that evaluates to 1 if and only if such an injection exists, is denoted by*

$$\texttt{Subgraph}^{H,v_1,\ldots,v_m}_{V,s_1,\ldots,s_m} : \{0,1\}^{\binom{V}{2}} \to \{0,1\},$$

and we use the following shorthand notation for the sequence of boolean functions

$$\texttt{Subgraph}^{H}_{v_1,\ldots,v_m} := (\texttt{Subgraph}^{H,v_1,\ldots,v_m}_{[n],1,\ldots,m})_{n=1}^{\infty}.$$

Whenever $m = 0$, we refer to the subgraph containment problem as unrestricted, in which case the boolean function is simply denoted by $\texttt{Subgraph}^{H}_{V}$, and the sequence of boolean functions by $\texttt{Subgraph}^{H}$. On the other hand, when $m \geq 1$, we say that the problem is restricted.

2. *We define the restricted and unrestricted subgraph containment problems analogously for directed graphs, i.e., if $H = (V_H, A_H)$ is a directed graph and we have adjacency-matrix-query access to a directed graph $G = (V, A)$, we define the boolean function* $\texttt{Subgraph}_V^H : \{0,1\}^{\{(v,w):v,w\in V:v\neq w\}} \to \{0,1\}$*, and similarly in the restricted setting.*
3. *Finally, we define finding versions of these problems, which ask to output* 0 *if the subgraph H is not contained in G, and the injective map φ otherwise. Formally, in the undirected unrestricted setting, this is a relation* $\texttt{FindSubgraph}_V^H \subseteq \{0,1\}^{\binom{V}{2}} \times (\{0\} \cup \{\varphi : V_H \to V \text{ injective}\})$*, and similarly for the restricted and directed settings.*

We can now phrase many graph problems formally as subgraph containment problems. For instance, if we let H be an undirected cycle on three vertices, then the sequence of boolean functions $\texttt{Subgraph}^H$ represents the triangle finding problem, where we have the bounds $\mathsf{Q}(\texttt{Subgraph}^H) \in O(n^{5/4}) \cap \Omega(n)$ [20, 11].

In this work, we restrict our attention to subgraph-containment problems where the subgraph is of constant size. In this setting, we can prove several elementary structural properties of these problems. The proofs of these properties are available in the full version of the paper.

▶ **Lemma 21.** *Let H be a graph of constant size, $m \in \mathbb{N}_0$ and $v_1, \dots, v_m \in V_H$. Suppose that $H_1, \dots, H_\ell$ are the connected components of H, and we write $i_1^{\ell'}, \dots, i_{m_{\ell'}}^{\ell'} \in [m]$ such that $v_{i_1^{\ell'}}, \dots, v_{i_{m_{\ell'}}^{\ell'}}$ are the vertices from $v_1, \dots, v_m$ that are contained in $H_{\ell'}$, for all $\ell' \in [\ell]$. Then, for $\mathsf{M} \in \{\mathsf{Q}, \mathsf{R}\}$, we have*

$$\mathsf{M}\left(\texttt{Subgraph}^H_{v_1,\dots,v_m}\right) \in \Theta\left(\max_{\ell'\in[\ell]}\left\{\mathsf{M}\left(\texttt{Subgraph}^{H_{\ell'}}_{v_{i_1^{\ell'}},\dots,v_{i_{m_{\ell'}}^{\ell'}}}\right)\right\}\right).$$

The above lemma informs us that we can without loss of generality focus on the case where H is connected. Next, we provide a full classification of these problems in the randomized setting, and a similar partial classification in the quantum setting. This lemma is folklore (see for instance [6, Proposition 4] for the third item), but we provide a formal statement and proof here for completeness.

▶ **Lemma 22.** *Let H be a connected non-empty graph of constant size, and let $m \in \mathbb{N}_0$ with $v_1, \dots, v_m \in V_H$. Then,*

1. *If $\{v_1, \dots, v_m\} = V_H$, then $\mathsf{Q}(\texttt{Subgraph}^H_{v_1,\dots,v_m}) \in \Theta(1)$ and $\mathsf{R}(\texttt{Subgraph}^H_{v_1,\dots,v_m}) \in \Theta(1)$.*
2. *If $H - \{v_1, \dots, v_m\}$ contains just isolated vertices, then $\mathsf{Q}(\texttt{Subgraph}^H_{v_1,\dots,v_m}) \in \Theta(\sqrt{n})$ and $\mathsf{R}(\texttt{Subgraph}^H_{v_1,\dots,v_m}) \in \Theta(n)$.*
3. *If $H - \{v_1, \dots, v_m\}$ contains an edge, then $\mathsf{Q}(\texttt{Subgraph}^H_{v_1,\dots,v_m}) \in \Omega(n)$ and we have $\mathsf{R}(\texttt{Subgraph}^H_{v_1,\dots,v_m}) \in \Theta(n^2)$.*

Note that the above lemma completely characterizes the randomized query complexity of all detection versions of undirected subgraph containment problems. The simplicity of this characterization is in stark contrast with the quantum setting, where such a characterization seems elusive.

Next, we provide some generic reductions between the detection and finding versions of the subgraph containment problems, and between the undirected and directed versions.

▶ **Lemma 23.** *Let H be a connected non-empty graph of constant size, $m \in \mathbb{N}_0$, and $v_1, \dots, v_m \in V_H$. Then, for $\mathsf{M} \in \{\mathsf{Q}, \mathsf{R}\}$, we have the characterization $\mathsf{M}(\texttt{Subgraph}^H_{v_1,\dots,v_m}) \in \Theta(\mathsf{M}(\texttt{FindSubgraph}^H_{v_1,\dots,v_m}))$.*

▶ **Lemma 24.** *Let H be a connected non-empty undirected graph of constant size, $m \in \mathbb{N}_0$, and $v_1, \ldots, v_m \in V_H$. Let $\vec{H}$ be the same graph as H, but then with arcs pointed in arbitrary directions. Then, for $\mathsf{M} \in \{\mathsf{Q}, \mathsf{R}\}$, we have $\mathsf{M}(\mathtt{Subgraph}^{H}_{v_1,\ldots,v_m}) \in O(\mathsf{M}(\mathtt{Subgraph}^{\vec{H}}_{v_1,\ldots v_m}))$.*

We note that the above lemmas also completely characterize the randomized query complexity for all directed and finding subgraph containment problems. We spend the rest of the document investigating the quantum counterparts of these results in more detail.

Finally, we recall that Theorem 1, as proved by Lee, Magniez and Santha, strictly speaking only holds for undirected subgraph-containment problems. However, we argue that their learning-graph-based algorithm also works in the directed setting, with the same complexity.

▶ **Proposition 25.** *Let $H = (V, A)$ be a directed graph, and let $H' = (V, E')$ be its undirected counterpart, i.e., $\{v, w\} \in E'$ if and only if $(v, w) \in A$ or $(w, v) \in A$. Then, $\mathsf{Q}(\mathtt{Subgraph}^{H})$ is upper bounded by the expression in Theorem 1 applied to H'.*

3.1 Path-containment reductions

We introduce shorthand notations for the subgraph-containment problems where the subgraph is a line graph. We refer to these as path-containment problems.

▶ **Definition 26** (Path-containment problems)**.** *Let $k \in \mathbb{N}$.*

1. *Let L_k be the undirected line graph with k edges, with the endpoint vertices labeled by s and t. We define $\mathtt{Path}^{=k} := \mathtt{Subgraph}^{L_k}$, and $\mathtt{Path}^{=k}_{s,t} := \mathtt{Subgraph}^{L_k}_{s,t}$.*
2. *Let L'_k be the directed version of L_k, where the edges are all pointing from s to t. We define $\mathtt{DirPath}^{=k} := \mathtt{Subgraph}^{L'_k}$, and $\mathtt{DirPath}^{=k}_{s,t} := \mathtt{Subgraph}^{L'_k}_{s,t}$.*

Finally, for $\mathtt{P} \in (|\mathtt{Dir})\mathtt{Path}_{(|s,t)}$, we write

$$\mathtt{P}^{\leq k} := \bigvee_{\ell=1}^{k} \mathtt{P}^{=\ell}.$$

We also consider the finding versions of these problems, denoted by $\mathtt{FindP}$ for all $\mathtt{P} \in (|\mathtt{Dir})\mathtt{Path}^{(=k|\leq k)}_{(|s,t)}$. In the $(\leq k)$-problems, it suffices to output a path for any of the subproblems $\mathtt{P}^{=\ell}$, with $\ell \in \{1, \ldots, k\}$.

We also consider promise versions of the path-containment problems.

▶ **Definition 27** (Promise path-containment problems)**.** *Let $k \in \mathbb{N}$.*

1. *The $\mathtt{PromPath}^{(=k|\leq k)}_{s,t}$-problem is the $\mathtt{Path}^{(=k|\leq k)}_{s,t}$-problem, restricted to inputs that satisfy the promise that if s and t are connected, then there exists a path of length exactly, or at most, k between s and t, respectively.*
2. *The $\mathtt{PromDirPath}^{(=k|\leq k)}_{s,t}$-problem is the $\mathtt{DirPath}^{(=k|\leq k)}_{s,t}$-problem, restricted to inputs that satisfy the promise that if s and t are connected by a directed path from s to t, then there exists a directed path of length exactly, or at most, k from s to t, respectively.*

The finding versions of these problems, $\mathtt{PromFindPath}^{(=k|\leq k)}_{s,t}$ and $\mathtt{PromFindDirPath}^{(=k|\leq k)}_{s,t}$ respectively, are the finding problems under the same restriction.

Formally speaking the promise path-containment problems are also boolean functions, but on a restricted domain. As such, they are partial booelan functions, rather than total ones.

We start by showing some elementary reductions between the promise and non-promise versions of the path-containment problems. The proofs are available in the full version of the paper.

▶ **Lemma 28.** *Let* $\mathtt{P} \in (|\mathtt{Dir})\mathtt{Path}_{s,t}^{(=k|\leq k)}$.
1. *We have* $\mathsf{Q}(\mathtt{PromP}) \in O(\mathsf{Q}(\mathtt{P}))$.
2. *We have* $\mathsf{Q}(\mathtt{FindP}) \in \Theta(\mathsf{Q}(\mathtt{PromFindP}))$.

Through Lemmas 23 and 28, we conclude that the `Find` and `PromFind` versions of the path-containment problems considered above all have the same query complexity up to constants as the non-promise detection version of the problem. Thus, to understand the relative hardness of the path-containment problems introduced above, we will ignore the `Find` and `PromFind` versions in what follows. We provide a graphical overview of the remaining problems in Figure 3, with arrows representing the reductions we prove in the remainder of this section.

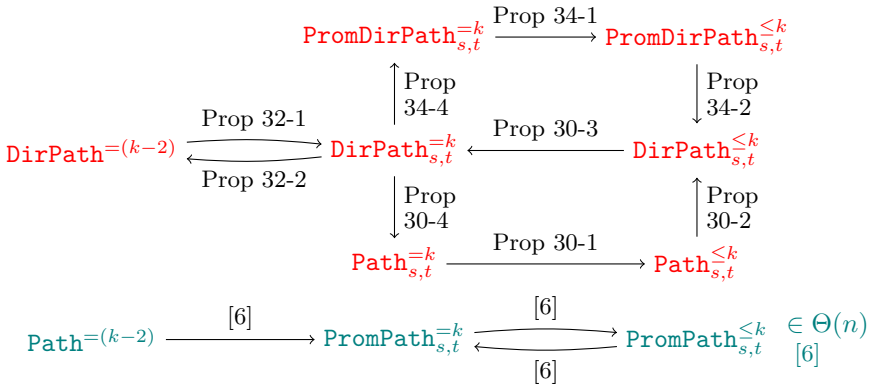


Figure 3 Randomized reductions between path-containment problems. All the green problems can be solved in linear number of queries, and all the red problems are equivalent under randomized reductions and constant multiplicative overhead.

If we set $k = 1$, or $k = 2$ then the problems $(|\mathtt{Dir})\mathtt{Path}_{s,t}^{(=k|\leq k)}$ are instances of the first and second classes derived in Lemma 22, respectively. Thus, whenever we set $k = 1$ or $k = 2$, we already understand the exact query complexity of the problems in Figure 3, and so we assume without loss of generality that $k \geq 3$.

For the $\mathtt{PromPath}_{s,t}^{\leq k}$-problem, Belovs and Reichardt [6] proved that it can be solved using $O(n\sqrt{k})$ queries, which is simply $O(n)$ when k is constant. They also realize that the $\mathtt{PromPath}_{s,t}^{=k}$-problem has a stronger promise, and so there is a generic reduction from this problem to the $\leq k$-version. Moreover, they provide a randomized reduction from the $\mathtt{Path}^{=(k-2)}$-problem to the $\mathtt{PromPath}_{s,t}^{=k}$-problem [6, Page 2]. This means that these three problems have linear query complexity, i.e., their query complexity is $\Theta(n)$ for $k \geq 3$.

We now prove that all the other path-containment problems form an equivalence class, i.e., they have the same query complexity up to a constant. We then prove a new quantum algorithm that solves all the problems from this equivalence class in Section 4.

We begin with showing monotonicity of $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{\leq k})$ in terms of k. From our results later in this section, this monotonicity result also follows for all the problems in the same equivalence class as $\mathtt{DirPath}_{s,t}^{\leq k}$. All proofs are available in the full version of the paper.

▶ **Lemma 29.** *Let* $3 \leq \ell \leq k \in \mathbb{N}$. *Then,* $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=\ell}) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k}))$.

▶ **Proposition 30.** *Let* $3 \leq k \in \mathbb{N}$. *Then,*
1. $\mathsf{Q}(\mathtt{Path}_{s,t}^{=k}) \in O(\mathsf{Q}(\mathtt{Path}_{s,t}^{\leq k}))$;
2. $\mathsf{Q}(\mathtt{Path}_{s,t}^{\leq k}) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{\leq k}))$;
3. $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{\leq k})) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k}))$;
4. $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k})) \in O(\mathsf{Q}(\mathtt{Path}_{s,t}^{=k}))$.

An immediate corollary of the above theorem is as follows.

▶ **Corollary 31.** *Let $3 \leq k \in \mathbb{N}$. Then, all the problems in $(|\mathtt{Dir})\mathtt{Path}_{s,t}^{(=k|\leq k)}$ have the same, up to constants, quantum query complexity.*

▶ **Proposition 32.** *Let $k \in \mathbb{N}$. Then,*
1. $\mathsf{Q}(\mathtt{DirPath}^{=k}) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k+2}))$;
2. *For $k \geq 3$, $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k}) \in O(\mathsf{Q}(\mathtt{DirPath}^{=k-2}))$.*

▶ **Corollary 33.** *Let $3 \leq k \in \mathbb{N}$. Then, the problems in $\mathtt{DirPath}_{s,t}^{=k}$ and $\mathtt{DirPath}^{=k-2}$ have the same, up to constants, quantum query complexity.*

▶ **Proposition 34.** *Let $3 \leq k \in \mathbb{N}$. Then,*
1. $\mathsf{Q}(\mathtt{PromDirPath}_{s,t}^{=k}) \in O(\mathsf{Q}(\mathtt{PromDirPath}_{s,t}^{\leq k}))$;
2. $\mathsf{Q}(\mathtt{PromDirPath}_{s,t}^{\leq k}) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{\leq k}))$;
3. $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{\leq k})) \in O(\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k}))$;
4. $\mathsf{Q}(\mathtt{DirPath}_{s,t}^{=k})) \in O(\mathsf{Q}(\mathtt{PromDirPath}_{s,t}^{=k}))$.

An immediate corollary of the above theorem is as follows.

▶ **Corollary 35.** *Let $3 \leq k \in \mathbb{N}$. Then, all the problems in $(|\mathtt{Prom})\mathtt{DirPath}_{s,t}^{(=k|\leq k)}$ have the same, up to constants, quantum query complexity.*

3.2 Cycle-containment reductions

We now introduce cycle-containment problems.

▶ **Definition 36** (Cycle-containment problems)**.** *Let $3 \leq k \in \mathbb{N}$.*
1. *Let C_k be the undirected cycle graph with k edges, and let s be any vertex on the cycle. Then, $\mathtt{Cycle}^{=k} := \mathtt{Subgraph}^{C_k}$ and $\mathtt{Cycle}_s^{=k} := \mathtt{Subgraph}_s^{C_k}$.*
2. *Let C'_k be the directed cycle graph with k arcs all pointing in the same direction, and let s be any vertex on the cycle. Then, $\mathtt{DirCycle}^{=k} := \mathtt{Subgraph}^{C'_k}$ and $\mathtt{DirCycle}_s^{=k} := \mathtt{Subgraph}_s^{C'_k}$.*

We also consider versions of the problem where we are looking for cycles of length at most k, i.e., for all $\mathtt{P} \in (|\mathtt{Dir})\mathtt{Cycle}_{(|s)}$, we write

$$\mathtt{P}^{\leq k} = \bigvee_{\ell=3}^{k} \mathtt{P}^{=k}.$$

Finally, we also consider the finding versions of these problems. That is, for all $\mathtt{P} \in (|\mathtt{Dir})\mathtt{Cycle}_{(|s)}^{(=k|\leq k)}$, we consider $\mathtt{FindP}$ where the task is to output the cycle. For the $(\leq k)$-problem, outputting a cycle of any length $3 \leq \ell \leq k$ suffices.

For simplicity, we only consider cycles of length at least 3. Technically, one can also have a "cycle" of length 2 in the directed setting, if two arcs in opposite directions are present between two vertices, but we will refrain from calling these cycles in this work.

We also consider promise problems of these problems.

▶ **Definition 37** (Promise cycle-containment problems)**.** *Let $3 \leq k \in \mathbb{N}$.*
1. *The $\mathtt{PromCycle}^{(=k|\leq k)}$ problem is the $\mathtt{Cycle}^{(=k|\leq k)}$-problem, restricted to inputs where if there exists any cycle in the graph, there must be at least one cycle of length exactly, or at most, k, respectively.*

2. *The* $\texttt{PromDirCycle}^{(=k|\leq k)}$ *problem is the* $\texttt{DirCycle}^{(=k|\leq k)}$*-problem, restricted to inputs where if there exists any directed cycle in the graph, there must be at least one directed cycle of length exactly, or at most,* k*, respectively.*
3. *The* $\texttt{PromCycle}_s^{(=k|\leq k)}$ *problem is the* $\texttt{Cycle}_s^{(=k|\leq k)}$*-problem, restricted to inputs where if there exists any cycle in the graph that passes through* s*, there must be at least one cycle of length exactly, or at most,* k*, respectively, that passes through* s*.*
4. *The* $\texttt{PromDirCycle}_s^{(=k|\leq k)}$ *problem is the* $\texttt{DirCycle}_s^{(=k|\leq k)}$*-problem, restricted to inputs where if there exists any directed cycle in the graph that passes through* s*, there must be at least one directed cycle of length exactly, or at most,* k*, respectively, that passes through* s*.*

We consider the finding versions of these problems as well, which are defined to be the finding problems under the same restrictions on the input, denoted by $\texttt{PromFind}(|\texttt{Dir})\texttt{Cycle}_{(|s)}^{(=k|\leq k)}$*.*

We note here that the same reductions hold between the promise and non-promise versions of the cycle-containment problems as in the path case.

▶ **Lemma 38.** *Let* $3 \leq k \in \mathbb{N}$*, and* $\texttt{P} \in (|\texttt{Dir})\texttt{Cycle}_{(|s)}^{(=k|\leq k)}$*.*
1. *We have* $\mathsf{Q}(\texttt{PromP}) \in O(\mathsf{Q}(\texttt{P}))$*.*
2. *We have* $\mathsf{Q}(\texttt{FindP}) \in \Theta(\mathsf{Q}(\texttt{PromFindP}))$*.*

Similar to the path-containment setting, the above lemma informs us that we don't need to consider the finding versions of the cycle-containment problems, since they are always equivalent to the non-promise detection versions up to randomized reductions. Thus, for all $\texttt{P} \in (|\texttt{Dir})\texttt{Cycle}_{(|s)}^{(=k|\leq k)}$, we only consider the problems $\texttt{P}$ and $\texttt{PromP}$ in the remainder of this section.

We now prove several randomized reductions between these problems, and we give a concise graphical overview of them in Figure 4.

For the following problems we can show that the quantum query complexity is monotonically increasing in k. The proofs are available in the full version of the paper.

▶ **Lemma 39.** *Let* $3 \leq \ell \leq k \in \mathbb{N}$*. Then,*
1. $\mathsf{Q}(\texttt{DirCycle}^{=\ell}) \in O(\mathsf{Q}(\texttt{DirCycle}^{=k}))$*;*
2. $\mathsf{Q}(\texttt{PromDirCycle}^{=\ell}) \in O(\mathsf{Q}(\texttt{PromDirCycle}^{=k}))$*;*
3. $\mathsf{Q}(\texttt{Cycle}^{\leq(k-1)}) \in O(\mathsf{Q}(\texttt{Cycle}^{\leq k}))$ *whenever* k *is an odd integer.*

The same monotonicity relations hold for $\texttt{P}_s$ *where* $\texttt{P} \in \{\texttt{DirCycle}, \texttt{PromDirCycle}, \texttt{Cycle}\}$*.*

We will now prove a few cycle-containment reductions as illustrated in Figure 4.

▶ **Proposition 40.** *Let* $3 \leq k \in \mathbb{N}$ *be a constant. Then,*
1. $\mathsf{Q}(\texttt{DirCycle}^{=k}) \in \Theta(\mathsf{Q}(\texttt{DirCycle}^{\leq k}))$*;*
2. $\mathsf{Q}(\texttt{PromDirCycle}^{\leq k}) \in \Theta(\mathsf{Q}(\texttt{PromDirCycle}^{=k}))$*;*
3. $\mathsf{Q}(\texttt{Cycle}^{\leq k}) \in O(\mathsf{Q}(\texttt{Cycle}^{=k}))$*;*
4. $\mathsf{Q}(\texttt{PromCycle}^{=k}) \in \Theta(\mathsf{Q}(\texttt{PromCycle}^{\leq k}))$*;*
5. $\mathsf{Q}(\texttt{DirCycle}^{=k}) \in O(\mathsf{Q}(\texttt{Cycle}^{=k}))$ *whenever* k *is an odd integer;*
6. $\mathsf{Q}(\texttt{PromCycle}^{\leq k}) \in O(\mathsf{Q}(\texttt{Cycle}^{\leq k}))$*;*
7. $\mathsf{Q}(\texttt{PromDirCycle}^{\leq k}) \in O(\mathsf{Q}(\texttt{DirCycle}^{\leq k})))$*;*
8. $\mathsf{Q}(\texttt{PromCycle}^{=k}) \in O(\mathsf{Q}(\texttt{PromDirCycle}^{=k}))$*;*
9. $\mathsf{Q}(\texttt{Cycle}^{\leq k}) \in O(\mathsf{Q}(\texttt{DirCycle}^{\leq k}))$*;*
10. $\mathsf{Q}(\texttt{Cycle}^{=k}) \in O(\mathsf{Q}(\texttt{DirCycle}^{=k}))$*.*

▶ **Theorem 41** (Implicit in [6, Theorem 8])**.** *Let* $3 \leq k \in \mathbb{N}$ *be a constant. Then, there exists a* $O(n)$*-query quantum algorithm that, given a simple graph* G *with* n *vertices, accepts if* G *contains a cycle of length* k *and rejects if* G *is a forest, except with error probability at most* $1/3$*.*

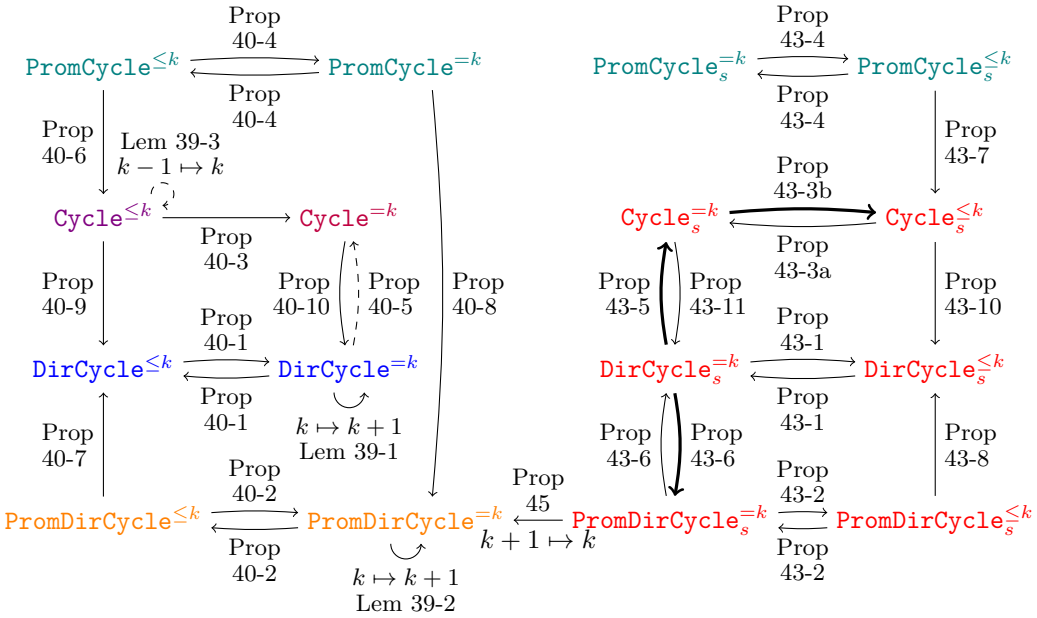


Figure 4 Pictorial representation of the randomized reductions we prove in this section, for $k \geq 3$. The dashed connections only hold for odd values of k. Problems depicted in the same color have the same quantum query complexity up to constants. Note that the connections represented in bold only exist in the restricted cycle-containment version of the problem, i.e., the corresponding result cannot be obtained in the left-hand side of the picture using the same techniques.

Combining Item 4 of Proposition 40 and Theorem 41 we conclude the following.

▶ **Corollary 42.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then,* $\mathsf{Q}(\texttt{PromCycle}^{=k|\leq k}) \in \Theta(n)$.

We will now present a few other cycle-containment reductions where the cycles are required to pass through a specific vertex, let us denote by s, given as input. For these reductions we will use color-coding and layer insertion techniques as done earlier, but we will use the color-coding version that passes through s. It is interesting to note that for some of these problems we are able to prove stronger relation in contrast to its counterparts that don't have this additional requirement of the subgraph containing s. For example, Items 3b and 5 in Proposition 43 are to name a few. See Figure 4 for an illustrative and comparative summary of the results.

▶ **Proposition 43.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then,*

1. $\mathsf{Q}(\texttt{DirCycle}_s^{=k}) \in \Theta(\mathsf{Q}(\texttt{DirCycle}_s^{\leq k}))$;
2. $\mathsf{Q}(\texttt{PromDirCycle}_s^{\leq k}) \in \Theta(\mathsf{Q}(\texttt{PromDirCycle}_s^{=k}))$;
3. **a.** $\mathsf{Q}(\texttt{Cycle}_s^{\leq k}) \in O(\mathsf{Q}(\texttt{Cycle}_s^{=k}))$;
   b. $\mathsf{Q}(\texttt{Cycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{Cycle}_s^{\leq k}))$;
4. $\mathsf{Q}(\texttt{PromCycle}_s^{=k}) \in \Theta(\mathsf{Q}(\texttt{PromCycle}_s^{\leq k}))$;
5. $\mathsf{Q}(\texttt{DirCycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{Cycle}_s^{=k}))$;
6. $\mathsf{Q}(\texttt{PromDirCycle}_s^{=k}) \in \Theta(\mathsf{Q}(\texttt{DirCycle}_s^{=k}))$;
7. $\mathsf{Q}(\texttt{PromCycle}_s^{\leq k}) \in O(\mathsf{Q}(\texttt{Cycle}_s^{\leq k}))$;
8. $\mathsf{Q}(\texttt{PromDirCycle}_s^{\leq k}) \in O(\mathsf{Q}(\texttt{DirCycle}_s^{\leq k}))$;
9. $\mathsf{Q}(\texttt{PromCycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{PromDirCycle}_s^{=k}))$;
10. $\mathsf{Q}(\texttt{Cycle}_s^{\leq k}) \in O(\mathsf{Q}(\texttt{DirCycle}_s^{\leq k}))$;
11. $\mathsf{Q}(\texttt{Cycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{DirCycle}_s^{=k}))$.

Consequently, we get the following.

▶ **Corollary 44.** *Let $3 \leq k \in \mathbb{N}$. Then, all the problems in* $(|\texttt{Prom})\texttt{DirCycle}_s^{(=k|\leq k)}$ *have the same, up to constants, quantum query complexity.*

Finally, we prove a connection between the restricted and unrestricted version of the cycle-containment problems.

▶ **Proposition 45.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then, we have* $\mathsf{Q}(\texttt{PromDirCycle}_s^{=(k+1)}) \in O(\mathsf{Q}(\texttt{PromDirCycle}_s^{=k}))$.

3.3 Path-cycle-containment reductions

In this subsection, we present reductions between path and cycle problems (in both directions) to further refine the classification established in Sections 3.1 and 3.2 for path and cycle problems, respectively. All the proofs are available in the full version of the paper.

▶ **Proposition 46.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then,* $\mathsf{Q}(\texttt{DirCycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{DirPath}_{s,t}^{=k}))$.

▶ **Proposition 47.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then, we have* $\mathsf{Q}(\texttt{PromDirPath}_{s,t}^{=k}) \in O(\mathsf{Q}(\texttt{PromDirCycle}_s^{=k}))$.

Using Corollary 35 from Section 3.1, Corollary 44 from Section 3.2 and Propositions 46 and 47 we conclude the following.

▶ **Corollary 48.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then, the path problems* $(|\texttt{Prom})\texttt{DirPath}_{s,t}^{(=k|\leq k)}$ *and the cycle problems* $(|\texttt{Prom})\texttt{DirCycle}_s^{(=k|\leq k)}$ *have the same, up to constants, quantum query complexity.*

▶ **Proposition 49.** *Let $3 \leq k \in \mathbb{N}$ be constant. Then,* $\mathsf{Q}(\texttt{PromCycle}_s^{=k}) \in O(\mathsf{Q}(\texttt{PromPath}_{s,t}^{=k}))$.

▶ **Proposition 50.** *Let $3 \leq k \in \mathbb{N}$ be constant. Then,* $\mathsf{Q}(\texttt{PromPath}_{s,t}^{=k}) \in O(\mathsf{Q}(\texttt{PromCycle}_s^{=k}))$.

As $\texttt{PromPath}_{s,t}^{=k}$ is known to have quantum query complexity of $\Theta(n)$ (see Figure 3), using Item 4 of Proposition 43 and Propositions 49 and 50 we can conclude the following:

▶ **Corollary 51.** *Let $3 \leq k \in \mathbb{N}$ be a constant. Then,* $\mathsf{Q}(\texttt{PromCycle}_s^{(=k|\leq k)}) \in \Theta(n)$.

4 Length-k directed path-detection algorithm

In this section, we consider the subgraph containment problem $\texttt{DirPath}^{=k}$, and we develop a novel algorithm for this problem. This implies a query upper bound for all of the problems displayed in red in Figure 1.

We start by employing the color-coding technique, introduced by Alon, Yuster and Zwick [1], to partition the graph into $k+1$ subsets. All proofs are available in the full version of the paper.

▶ **Lemma 52.** *Let $k, n \in \mathbb{N}$, and let $G = (V, A)$ be a directed graph with $|V| = n$ and $k \in O(1)$. Let $V_0 \sqcup \cdots \sqcup V_k \subseteq V$ be a random partition of V such that $|V_j| \in \Theta(n/k)$. We define $G' = (V, A')$, where $A' = A_1 \sqcup \cdots \sqcup A_k$, with $A_j = \{(v, w) \in A : v \in V_{j-1}, w \in V_j\}$. Now, if $\texttt{DirPath}^{=k}(G) = 0$, then $\texttt{DirPath}^{=k}(G') = 0$. On the other hand, if $\texttt{DirPath}^{=k}(G) = 1$, then $\mathbb{P}_{G'}[\texttt{DirPath}^{=k}(G') = 1] \in \Omega(1)$, where the probability distribution is over the randomization in the procedure that constructs G'.*

We can combine the previous theorem with Lemma 17, so without loss of generality we can consider the color-coded version of the problem, i.e., where the graph's vertices are already partitioned into $k+1$ disjoint sets and where we're looking for a path that traverses all the way from one end to the other, as displayed in Figure 5. To optimize the parameters in the recursion, we consider a version this problem where the first and last layer have a smaller size than all the intermediate layers, and setting $r \in \Theta(n)$ recovers the problem considered in Lemma 52.

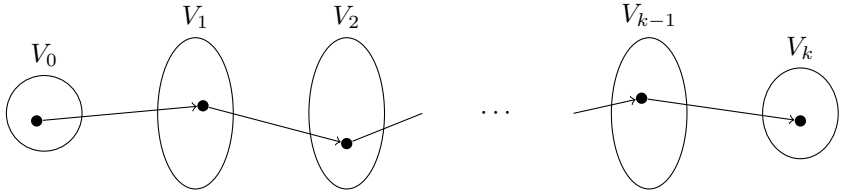


Figure 5 The layered-path problem. Every set V_j for $j \in [k-1]$ represents a set of $\Theta(n)$ vertices, and the first and last layer might have a different size, say $|V_0| = |V_k| = r$. We are looking for a directed path that traverses all the layers from V_0 to V_k.

We now provide a recursive nested walk construction that finds such a layered path, if it exists.

▶ **Lemma 53.** *Let $k, n, r \in \mathbb{N}$, with $3 \leq k \in O(1)$ and $r \leq n$. Let $n/2 \geq s \in \mathbb{N}$, and suppose that we can solve the layered path-finding problem with length $k-2$ and first and last layer size s with $C_{n,s}^{(k-2)}$ queries. Then, we can solve the layered path-finding problem with complexity $C_{n,r}^{(k)}$, satisfying*

$$C_{n,r}^{(k)} \in \widetilde{O}\left(s\sqrt{r} + \left(\frac{n}{s}\right)\left(\sqrt{sr} + C_{n,s}^{(k-2)}\right)\right).$$

Now that we have described the recursive algorithm, we should work out what the resulting number of queries becomes, for a given value of $k \in O(1)$. This is the objective of the following lemma.

▶ **Lemma 54.** *Let $\alpha_r \in [0,1]$. For all $k \in \mathbb{N}$, let $\alpha_{k,r}$ be defined by the recurrence relation*

$$\alpha_{k+2,r} := \min_{\alpha_s \in [0,1]} \max\left\{\alpha_s + \frac{1}{2}\alpha_r, 1 - \frac{1}{2}\alpha_s + \frac{1}{2}\alpha_r, 1 - \alpha_s + \alpha_{k,s}\right\},$$

with $\alpha_{1,r} = \alpha_r$, and $\alpha_{2,r} = \frac{1}{2} + \frac{1}{2}\alpha_r$. Next, let $n \in \mathbb{N}$, and $r \in \Theta(n^{\alpha_r})$. Then, we can solve the layered-path problem with first and last layer size r with a number of queries that satisfies $C_{n,r}^{(k)} \in \widetilde{O}(n^{\alpha_{k,r}})$. Moreover, the solution to the recurrence relation satisfies $\alpha_{k,n} \in 3/2 - \Theta(c^{-k})$ with $c \approx 1.33$.

Putting everything together, we now obtain the following result.

▶ **Corollary 55.** $\mathrm{Q}(\texttt{DirPath}^{=k}) \in \widetilde{O}(n^{\frac{3}{2}-\alpha_k})$, *where $\alpha_k \in \Omega(c^{-k})$ as $k \to \infty$, with $c = \sqrt{3+\sqrt{17}}/2 \approx 1.33$.*

We compute the resulting query complexity for the first few values of k for good measure.

5 Length-$\leq k$ cycle-detection algorithm

In this section, we modify the approach developed in [14], to obtain a novel quantum algorithm for the $\texttt{Cycle}^{\leq k}$-problem, whenever k is odd. The core result that we are using is the following theorem.

Table 1 The query complexity of $\mathsf{Q}(\texttt{DirPath}^{=k})$ for the first few values of k.

k	1	2	3	4	5	6	7
$\mathsf{Q}(\texttt{DirPath}^{=k})$	$\widetilde{O}(n)$	$\widetilde{O}(n)$	$\widetilde{O}(n^{7/6})$	$\widetilde{O}(n^{7/6})$	$\widetilde{O}(n^{23/18})$	$\widetilde{O}(n^{29/22})$	$\widetilde{O}(n^{91/66})$

▶ **Theorem 56** ([14, Theorem 4.7]). *Let $\mathcal{P}$ be the property that an n-vertex graph either has more than $\overline{m}$ edges, where $\overline{m} \in \Omega(n)$, or contains a given subgraph H. Let H' be the graph obtained by deleting all degree-1 vertices of H that are not part of an isolated edge. Then $\mathsf{Q}(\mathcal{P}) \in \widetilde{O}(\sqrt{\overline{m}}n^{1-\frac{1}{vc(H')+1}})$, where $vc(H')$ denotes the vertex cover of H', i.e., the minimum number of vertices in H' required to cover all edges.*

Applying this to k-cycles, we get the following corollary.

▶ **Corollary 57.** *Let $k \geq 3$. Let $\mathcal{P}$ be the property that an n-vertex graph either has more than $\overline{m}$ edges, where $\overline{m} \in \Omega(n)$, or contains a k-cycle. Then, there is a quantum algorithm that computes $\mathcal{P}$ using $\widetilde{O}(\sqrt{\overline{m}}n^{1-\frac{1}{\lceil k/2 \rceil+1}})$ queries.*

We modify the algorithm attaining the result in Theorem 56 (and Corollary 57) to get a slight improvement for the case when k is odd (making a stronger assumption on $\overline{m}$ which will also be satisfied for our applications). The proof is available in the full version of the paper.

▶ **Theorem 58.** *Let $k \geq 3$. Let $\mathcal{P}$ be the property that an n-vertex graph either has more than $\overline{m}$ edges, where $\overline{m} \in \Omega(n^{1+1/(\ell+1)})$ and $\ell = \lfloor k/2 \rfloor$, or contains a k-cycle. Then, there is a quantum algorithm SparseCycle that computes $\mathcal{P}$ using $\widetilde{O}(\sqrt{\overline{m}}n^{1-\frac{1}{\ell+1}})$ queries.*

Now, we can combine this construction with a combinatorial result by Bondy and Simonovits [7], that connects the non-existence of an even-length cycle of a given length to a sparsity condition on the graph.

▶ **Theorem 59** ([7]). *Let G be a graph on n vertices. For any $\ell \geq 2$, if $|E(G)| > 100\ell n^{1+1/\ell}$ then G contains $\mathcal{C}_{2\ell}$ as a subgraph.*

Combining the previous two results now gives us our algorithm that solves the $\texttt{Cycle}^{\leq k}$-problem. The proof is available in the full version of the paper.

▶ **Theorem 60.** *Let $k \geq 4$. Then, $\mathsf{Q}(\texttt{Cycle}^{\leq k}) \in \widetilde{O}(n^{\frac{3}{2}-\gamma(k)})$, where*

$$\gamma(k) = \begin{cases} \frac{k-2}{k(k+2)}, & \text{if } k \text{ is even,} \\ \frac{k-3}{(k-1)(k+1)}, & \text{if } k \text{ is odd.} \end{cases}$$

Finally, we summarize the best-known upper bounds on the query complexities for $\texttt{Cycle}^{=k}$ and $\texttt{Cycle}^{\leq k}$ in Figure 6. We observe that for all k, the $\texttt{Cycle}^{\leq k}$-problem can be solved in $o(n^{3/2})$ queries, whereas for the $\texttt{Cycle}^{=k}$-problem, we can only make that statement for even values of k. It would be a very interesting direction of future research to establish whether the $\texttt{Cycle}^{=k}$-problem for odd values of k can be solved in $\widetilde{O}(n^{3/2})$ queries as well.

We also observe that the expressions we obtain for general values of k do not recover the best-known upper bounds for the cases where $k = 3$ and $k = 4$. This suggests that for higher values of k, the upper bounds we know for these cycle-containment problems are likely not tight. We think it's a very nice future direction of research to improve these algorithms.

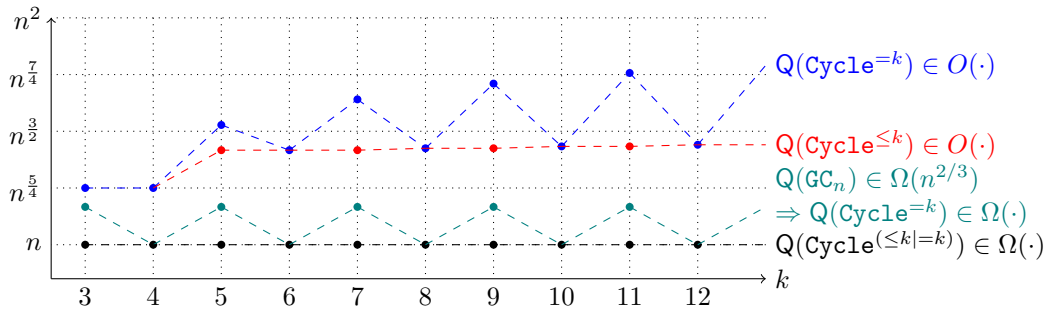


Figure 6 The best-known upper bounds on the query complexity of the $\texttt{Cycle}^{=k}$-problem (shown in blue) and the $\texttt{Cycle}^{\leq k}$-problem (shown in red). The black vertices represent the best-known unconditional lower bounds on both problems, and the green dots represent the best-known conditional lower bounds on the $\texttt{Cycle}^{=k}$-problem, conditional on $\mathsf{Q}(\texttt{GC}_n) \in \Omega(n^{2/3})$.

6 Conditional lower bounds based on graph collision

In this final section, we prove a novel super-linear conditional lower bound on the query complexities of several problems in Figure 1. Our lower bounds will be based on the graph-collision problem, introduced in [24, Section 4.2].

▶ **Definition 61** (Graph collision problem [24, Section 4.2]). *Let $G = (V, E)$ be an undirected graph. We consider the boolean function $\texttt{GC}_G = \texttt{GraphCollision}_G : \{0,1\}^V \to \{0,1\}$, which evaluates to 1 on input $x \in \{0,1\}^V$ if and only if there exists an edge $vw \in E$, such that $x_v = x_w = 1$. This is the graph collision problem on G. For all $n \in \mathbb{N}$,*

$$\mathsf{Q}(\texttt{GC}_n) := \max_{\substack{G=(V,E)\\|V|=n}} \mathsf{Q}(\texttt{GC}_G).$$

Balodis and Iraids [3] showed that the quantum query complexity of triangle finding is related to the graph collision problem, which gives us a conditional lower bound on several of the equivalence classes displayed in Figure 1. We modify Balodis's and Iraids's approach to give the same lower bound to the cycle-finding problem through a vertex s. The proof is available in the full version of the paper.

▶ **Proposition 62.** $\mathsf{Q}(\texttt{Cycle}_s^{=5}) \in \Omega(\sqrt{n} \cdot \mathsf{Q}(\texttt{GC}_n))$.

References

1 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *Journal of the ACM (JACM)*, 42(4):844–856, 1995. doi:10.1145/210332.210337.

2 Andris Ambainis. Quantum walk algorithm for element distinctness. *SIAM Journal on Computing*, 37(1):210–239, 2007. doi:10.1137/S0097539705447311.

3 Kaspars Balodis and Jānis Iraids. Quantum lower bound for graph collision implies lower bound for triangle detection. *Baltic Journal of Modern Computing*, 4(4), 2016. doi:10.22364/BJMC.2016.4.4.10.

4 Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, 1998.

5 Salman Beigi and Leila Taghavi. Quantum speedup based on classical decision trees. *Quantum*, 4:241, 2020. doi:10.22331/Q-2020-03-02-241.

6 Aleksandrs Belovs and Ben W Reichardt. Span programs and quantum algorithms for st-connectivity and claw detection. In *European Symposium on Algorithms*, pages 193–204. Springer, 2012. doi:10.1007/978-3-642-33090-2_18.

7 J. Adrian Bondy and Miklós Simonovits. Cycles of even length in graphs. *Journal of Combinatorial Theory, Series B*, 16(2):97–105, 1974.

8 Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *AMS Contemporary Mathematics Series*, 305, 2000.

9 Harry Buhrman, Ronald de Wolf, Christoph Dürr, Mark Heiligman, Peter H"yer, Frédéric Magniez, and Miklos Santha. Quantum algorithms for element distinctness. In *Proceedings of the 16th Annual Conference on Computational Complexity*, 2001.

10 Chris Cade, Ashley Montanaro, and Aleksandrs Belovs. Time and space efficient quantum algorithms for detecting cycles and testing bipartiteness. *Quantum Information and Computation*, 18(1-2):18–50, 2018. doi:10.26421/QIC18.1-2-2.

11 Titouan Carette, Mathieu Laurière, and Frédéric Magniez. Extended learning graphs for triangle finding. *Algorithmica*, 82(4):980–1005, 2020. doi:10.1007/S00453-019-00627-Z.

12 Andrew M. Childs, Richard Cleve, Enrico Deotto, Edward Farhi, Sam Gutmann, and Daniel A. Spielman. Exponential algorithmic speedup by a quantum walk. In *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*, 2003.

13 Andrew M. Childs, Matthew Coudron, and Amin Shiraz Gilani. Quantum Algorithms and the Power of Forgetting. In *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, 2023. doi:10.4230/LIPIcs.ITCS.2023.37.

14 Andrew M Childs and Robin Kothari. Quantum query complexity of minor-closed graph properties. *SIAM Journal on Computing*, 41(6):1426–1450, 2012. doi:10.1137/110833026.

15 Arjan Cornelissen. Quantum walks through generalized graph composition. *arXiv preprint*, 2025. doi:10.48550/arXiv.2510.04973.

16 Christoph Dürr, Mark Heiligman, Peter Hoyer, and Mehdi Mhalla. Quantum query complexity of some graph problems. *SIAM Journal on Computing*, 35(6):1310–1328, 2006. doi:10.1137/050644719.

17 Peter Hoyer, Troy Lee, and Robert Spalek. Negative weights make adversaries stronger. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, pages 526–535, 2007.

18 Stacey Jeffery. Quantum subroutine composition. *arXiv preprint*, 2022. doi:10.48550/arXiv.2209.14146.

19 Stacey Jeffery, Robin Kothari, and Frédéric Magniez. Nested quantum walks with quantum data structures. In *Proceedings of the twenty-fourth annual ACM-SIAM symposium on Discrete algorithms*, pages 1474–1485. SIAM, 2013. doi:10.1137/1.9781611973105.106.

20 François Le Gall. Improved quantum algorithm for triangle finding via combinatorial arguments. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 216–225. IEEE, 2014. doi:10.1109/FOCS.2014.31.

21 Troy Lee, Frédéric Magniez, and Miklos Santha. Learning graph based quantum query algorithms for finding constant-size subgraphs. *Chicago Journal of Theoretical Computer Science*, 10:1–21, 2012.

22 Cedric Yen-Yu Lin and Han-Hsuan Lin. Upper bounds on quantum query complexity inspired by the elitzur-vaidman bomb tester. In *Proceedings of the 30th Conference on Computational Complexity*, pages 537–566, 2015. doi:10.4230/LIPIcs.CCC.2015.537.

23 Frédéric Magniez, Ashwin Nayak, Jérémie Roland, and Miklos Santha. Search via quantum walk. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 575–584, 2007. doi:10.1145/1250790.1250874.

24 Frédéric Magniez, Miklos Santha, and Mario Szegedy. Quantum algorithms for the triangle problem. *SIAM Journal on Computing*, 37(2):413–424, 2007. doi:10.1137/050643684.

25 Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

26 Robert Špalek and Mario Szegedy. All quantum adversary methods are equivalent. *Theory of Computing*, 2(1):1–18, 2006. doi:10.4086/TOC.2006.V002A001.

27 Shengyu Zhang. On the power of ambainis lower bounds. *Theoretical Computer Science*, 339(2-3):241–256, 2005. doi:10.1016/J.TCS.2005.01.019.

28 Yechao Zhu. Quantum query complexity of subgraph containment with constant-sized certificates. *arXiv preprint*, 2011. arXiv:1109.4165.

Testing the Independent Set Property in Hypergraphs

Elena Grigorescu
Cheriton School of Computer Science, University of Waterloo, Canada

Shreya Nasa[1]
Department of Computer Science, Purdue University, West Lafayette, IN, USA

Cameron Seth
Cheriton School of Computer Science, University of Waterloo, Canada

Abstract

The optimal sample complexity of testing if an n-vertex graph has an independent set of size ρn, or is ε-far from having an independent set of size ρn, was established to be $\widetilde{O}(\rho^3/\varepsilon^2)$, in a notable result by Blais and Seth (SICOMP 2025). In contrast, for q-uniform hypergraphs, there is a significant gap between the best known upper and lower bounds, and there has been no progress on the problem for the last two decades. In this work, we prove a new upper bound of $\widetilde{O}\left(\frac{q\rho^{2q-3}}{\varepsilon^2(q-2)!^2}\right)$ on the sample complexity of testing the ρ-independent set property. The previous best known upper bound was $\widetilde{O}\left(\frac{2^q q!\rho^{2q}}{\varepsilon^3}\right)$, due to Langberg (RANDOM 2004). This establishes the optimal dependence on ε and gives an exponential improvement in the dependence on q. We prove our result via a new application of the hypergraph container method.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Randomness, geometry and discrete structures; Theory of computation → Theory and algorithms for application domains

Keywords and phrases independent set, property testing, hypergraph container method, hypergraph

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.73

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2607.13011`

Acknowledgements We would like to thank Eric Blais for his useful insights and suggestions throughout the project.

1 Introduction

Graph and hypergraph property testing address the question of when one can distinguish between the case that a (hyper)graph has some property and the case that it is *far* from having the property, by inspecting only a small fraction of the input (hyper)graph. In this work we study the classic problem of testing whether a hypergraph has a large independent set in the dense hypergraph model. We say that a q-uniform hypergraph H on n vertices has the ρ-independent set property if H has an independent set I of size ρn, and H is considered to be ε-far from having a ρn independent set if at least εn^q edges must be removed from H to obtain a hypergraph with an independent set of size ρn.

An ε-*tester* for ρ-independence is a bounded error randomized algorithm that takes a random sample $S \subseteq V$ of vertices, inspects $H[S]$, and, if H has a ρn independent set, accepts with probability at least δ, and if H is ε-far from having a ρn independent set, accepts with

[1] Supported in part by NSF Awards CCF-2127806, CCF-2228814, and ONR Award N00014-24-1-2695

© Elena Grigorescu, Shreya Nasa , and Cameron Seth;
licensed under Creative Commons License CC-BY 4.0

probability at most γ, for some positive constants $\delta > \gamma$.[2] The sample size $|S|$ is known as the sample complexity of the tester, and the sample complexity of testing the ρn independent set property is the minimum sample complexity of any such ε-tester for the property.

Independent set testing on graphs (when $q = 2$) is one of the classic graph properties which was studied by Goldreich, Goldwasser and Ron when they introduced the notion of graph property testing [6]. They showed that it is possible to ε-test the ρ-independent set property in graphs with sample complexity $\widetilde{O}(\rho/\varepsilon^4)$. Feige, Langberg and Schechtman later improved the bound to $\widetilde{O}(\rho^4/\varepsilon^3)$ [5]. The problem of finding the optimal sample complexity in graphs remained open for nearly 20 years until a recent work by Blais and Seth improved the bound to $\widetilde{O}(\rho^3/\varepsilon^2)$ by introducing a new connection between property testing and the graph container method [4]. The result by Blais and Seth is optimal up to logarithmic (in $1/\varepsilon$) factors based on a lower bound in [5].

Finding the optimal sample complexity of ρ-independent set testing in q-uniform hypergraphs remains open. Generalizing the results of [5], Langberg showed that there is an ε-tester for ρ-independent set testing in hypergraphs with sample complexity of $\widetilde{O}\left(\frac{2^q q! \rho^{2q}}{\varepsilon^3}\right)$ [8]. Additionally, Langberg gave a lower bound on the sample complexity of $\widetilde{\Omega}\left(\frac{\rho^{2q-1}}{(q!)^2\varepsilon^2}\right)$. This leaves the natural open question: what is the optimal dependence on ε for the sample complexity of testing ρ-independent sets in q-uniform hypergraphs? Further, there is also an exponential gap in the q dependence between the upper and lower bounds. It's worth noting that even though the upper bound has a ρ^{2q} in the numerator and the lower bound has a ρ^{2q-1} in the numerator, the problem is only non-trivial when $\varepsilon < \frac{\rho^q}{q!}$ and so the lower bound is strictly smaller than the upper bound.

1.1 Our Results

We give a new upper bound on the sample complexity of testing the ρ-independent set property in hypergraphs.

▶ **Theorem 1.** *The sample complexity for testing the ρ-independent set property is $\widetilde{O}\left(\frac{q\rho^{2q-3}}{\varepsilon^2(q-2)!^2}\right)$.*

Our analysis achieves the optimal dependence on ε, matching the lower bound of Langberg, while also improving the dependence on q exponentially compared to previous results.

There is a gap between our upper bound and the lower bound of Langberg of roughly $\frac{\rho^2}{q^5}$. We note that the lower bound result of Langberg can be improved slightly to $\frac{\rho^{2q-1}}{((q-1)!)^2\varepsilon^2}$ using the same proof approach (see Appendix A), and so this reduces the gap between upper and lower bounds to roughly $\frac{\rho^2}{q^3}$. Achieving the optimal ρ dependence remains an open problem, however we show in Remark 7 that our analysis cannot be tightened to achieve the optimal ρ-dependence.

We note that, while our bound applies for any q, in the regime that q is non-constant, there is an alternate distance measure. We could instead say a q-uniform hypergraph is γ-far from having a ρn independent set if at least $\gamma\binom{n}{q}$ edges must be modified to make the hypergraph have the property, similar to what was studied by [3] for satisfiability testing. Observe that $\gamma\binom{n}{q} \approx \frac{\gamma}{q!}n^q$, and so if a hypergraph is γ-far from having a ρn independent set

[2] The standard success acceptance probabilities are $\delta = 2/3$ and $\gamma = 1/3$, but so long as $\delta > \gamma$ the probabilities can be boosted to $2/3$ and $1/3$ by standard error reduction techniques. We note that if $\delta = 1$ the tester is called a one-sided error tester, and otherwise it's called a two-sided error tester.

under this alternate distance measure then it is $\varepsilon \approx \frac{\gamma}{q!}$-far in the distance measure we use in this paper. Using Theorem 1 with this change of variable we get an upper bound of roughly $\frac{q^5\rho^{2q-3}}{\gamma^2}$ for testing ρ-independence under this alternate notion of distance.

1.2 Overview of Techniques

To prove Theorem 1 we analyze the natural testing algorithm that takes a random sample $S \subseteq V$ of vertices and accepts if and only if $H[S]$ contains an independent set of size $\rho|S|$. This is a two-sided error tester, which is necessary for any tester for the ρ-independent property, as it is not a semihereditary property [1]. As is common for property testing algorithms, the main challenge is showing that the tester rejects hypergraphs that are ε-far from having a ρn independent set with high probability.

The main tool in our analysis is a new application of the hypergraph container method. Blais and Seth first applied the graph container method to analyze property testers for the ρ-independent set and k-colorability properties on graphs [4]. They further used the hypergraph container method to analyze the sample complexity for the satisfiability problem and hypergraph coloring [3]. We adapt their hypergraph container method algorithm to analyze the sample complexity of ρ-independence testing in hypergraphs. We first briefly describe the container method and the approach used by Blais and Seth for graphs, and then discuss the main challenges with adapting the container method to analyzing testers for the ρ-independent set property on hypergraphs.

1.2.1 The Container Method

The container method is a tool for characterizing independent sets in graphs with specific structure, commonly regular or nearly regular graphs. While in general a graph on n vertices may have up to 2^n independent sets, the container method can be used to show that for a graph G satisfying some specific structure, there is a relatively small collection of containers, where each container is a subset of vertices and each container is relatively small, such that every independent set is a subset of a container.

The natural way to construct the collection of containers is with a container generating algorithm, first used by Kleitman and Winston [7], which works as follows. Given a graph $G = (V, E)$ and an independent set $I \subseteq V$, the goal of the container generating algorithm is to find a small fingerprint $F \subseteq I$ that can be used to construct a small container $C \subseteq V$ such that $I \subseteq C$. In graphs, the natural approach is an iterative greedy approach that first initializes the candidate container to V, and in each step adds a vertex $v \in I$ to the fingerprint which has highest degree in the current container candidate set. Then, all the neighbours of v are removed from the current container because they cannot be in the independent set. Further, any vertex with degree higher than v in the current container can be removed as well because they also cannot be in the independent set (otherwise they would have been selected to the fingerprint). The container generating procedure stops once the container is sufficiently small. This procedure is run on every independent set to construct the collection, however, since each constructed fingerprint corresponds to a unique container, and each fingerprint is relatively small, then there are a small number of containers (corresponding to the total number of possible fingerprints).

1.2.2 Overview of Blais–Seth Approach

Blais and Seth showed that the container method applies very naturally to graphs that are ε-far from having a ρn independent set [4]. They showed that when G is ε-far from having a ρn independent set, there is a collection of containers, where each container is a subset of

vertices, such that every independent set is a subset of a container. They show that each container is small (strictly smaller than ρn) and that there are not too many of them (at most roughly $n^{1/\varepsilon}$).

Blais and Seth then use this collection of containers to analyze the natural testing algorithm that takes a random sample $S \subseteq V$ of vertices and checks if $G[S]$ has an independent set of size $\rho|S|$. In particular, when G is ε-far from having a ρn independent set, they use the collection of containers to upper bound the probability that $G[S]$ has a $\rho|S|$ independent set as follows. By the container statement, since every independent set is a subset of a container, $G[S]$ only has a $\rho|S|$ independent set if at least $\rho|S|$ vertices of S are drawn from a specific container. The probability that $\rho|S|$ vertices are drawn from a specific container can be upper bounded by a Chernoff bound, and then using a union bound they upper bound the probability that $\rho|S|$ vertices are drawn from any container, thereby upper bounding the probability that $G[S]$ has a $\rho|S|$ independent set.

In order to get the optimal bound for ρn independent set testing, Blais and Seth are crucially able to show that there is a trade off in the container and fingerprint size. In particular, they show that large containers have small fingerprints and vice versa. A key step in their proof is an argument that roughly says the following: given a graph $G = (V, E)$ that is ε-far from having an independent set and some independent set I, if the container generating procedure ends up at a very large (and sparse) container $C \subseteq V$, then either $|E(C, V \setminus C)|$ is relatively large, or $G[V \setminus C]$ is very dense. They show that in either case a large number of vertices would be removed from the container in a single step, and, further, this holds at every step of the container procedure. Since many vertices are removed in each step of the procedure then the procedure must terminate in a relatively small number of steps, resulting in a relatively small fingerprint.

1.2.3 First Challenge with Adapting to the Hypergraph Problem

One immediate challenge for hypergraphs is that the container generating procedure is not as simple as in the graph case. In graphs, selecting a fingerprint vertex to be the vertex in I with highest degree in the current container works very well because if $v \in I$ then the neighbours of v must not in the independent set and so can immediately be removed from the current container. However, even for 3-uniform hypergraphs this does not work: if $v \in I$ then vertices that share an edge with v may still be in the independent set.

We address this challenge using a similar approach as used by Blais and Seth in their follow-up work on testing satisfiability [3], and also in hypergraph container method literature (see [2, 10] for example). In particular, in each iteration of our container generating procedure, we have an inner loop which takes q-edges and turns them into $(q-1)$-edges, and repeats on the corresponding $(q-1)$-uniform hypergraph. For example, in the 3-uniform hypergraph case, if $v \in I$ then for any pair $\{u, w\}$ for which $\{v, u, w\}$ is an edge in H, at most one of u or w can be in I. So, after selecting a vertex $v \in I$ to the fingerprint, the container procedure will construct a graph (i.e. a 2-uniform hypergraph) where the edges are all pairs $\{u, w\}$ such that $\{v, u, w\}$ is an edge in H. The procedure then can operate on the graph as before by selecting another vertex in I and removing neighbours (see Section 2 for more details).

1.2.4 Second Challenge with Adapting to the Hypergraph Problem

The second key challenge in adapting the Blais–Seth approach to get nearly tight bounds for hypergraph ρ-independent set testing is that the argument about the trade off between the container and fingerprint size cannot be generalized directly to hypergraphs. As mentioned

above, they show that given a graph G that is ε-far from having an independent set and some independent set I, if the container procedure ends up at a very large (and sparse) container C, then either $E(C, V \setminus C)$ is relatively large, or $G[V \setminus C]$ is very dense. This step of the proof does not generalize naturally to hypergraphs because the edges between C and $V \setminus C$ can now take many forms (i.e. i endpoints in C and $(q-i)$ endpoints in $V \setminus C$ for any $1 \leq i \leq q-1$). In fact, using a standard hypergraph container generating procedure to analyze a tester for ρ-independent set would result in a sample complexity that is larger than our upper bound by at least roughly $1/\rho^q$ (see Remark 8 for an example to demonstrate this).

In order to get our tighter upper bound on the sample complexity we change the hypergraph container procedure following ideas from [3]. Instead of selecting a vertex with highest degree in the hypergraph to the fingerprint, we select a vertex to the fingerprint with highest degree in some specific ρn-subhypergraph. Then the inner loop focuses on this specific ρn-subhypergraph. We note that Blais and Seth [3] use a similar approach in the hypergraph container method by focusing on a smaller n-vertex subhypergraph in a kn-vertex graph for their CSP hypergraph construction.

We note that once we make this change to the container procedure, the proof approach is similar to that used by Blais and Seth to study k-colorability and satisfiability [3]. Roughly speaking the proof goes by contradiction. Given a hypergraph H that is ε-far from having a large independent set, and an independent set I, we assume that for all steps of the procedure the container is larger than desired. We prove a proposition which shows that the max degree in the container decreases in each step, and use this to reconstruct a ρn-subhypergraph that demonstrates that H is not ε-far from having a ρn independent set, a contradiction. Since we reconstruct a subhypergraph of size ρn to prove the contrapositive, we obtain a tighter bound by considering the maximum q-degree of a vertex in any ρn-induced subhypergraph in the container.

2 New Container Lemma

In this section, we prove a new container lemma for hypergraphs, which will be used in Section 3 to prove our upper bound $\widetilde{O}\left(\frac{q\rho^{2q-3}}{\varepsilon^2(q-2)!^2}\right)$ on the sample complexity. We start by describing our hypergraph container procedure and formalizing some basic properties of the fingerprints and the containers which is then used to prove the new container lemma for q-uniform hypergraphs. We note that Blais and Seth [3] use similar ideas to prove their upper bound on the sample complexity of CSP.

We defer the proof for the lower bound $\Omega\left(\frac{\rho^{2q-1}}{\varepsilon^2(q-1)!^2}\right)$ using the construction by Langberg [8] to Appendix A.

2.1 The Container Method

Our hypergraph container algorithm builds on the work of Kleitman and Winston [7] and of Sapozhenko [9]. The key idea behind the container method is to identify a small set of vertices called the *fingerprint* vertices which uniquely identify its *containers* (upto a predefined ordering of vertices) for an input independent set I in a q-uniform hypergraph $H = (V, E)$. We denote the fingerprint and container at t^{th} iteration of the algorithm as $F_t(I)$ and $C_t(F_t(I))$ respectively. We may use the shorthand notation F_t and $C_t(F_t)$ when the underlying independent set is clear from the context.

The algorithm (see Algorithm 1) starts with initializing the entire vertex set as the container C_0 and an empty set of vertices as the fingerprint F_0. We add vertices to the fingerprint and remove vertices from its corresponding container as the algorithm progresses. At t^{th} iteration, the algorithm chooses the vertex v_t from the container C_{t-1} as the vertex in the input independent set $I \subseteq V$ with the maximum q-degree in any ρn-induced subhypergraph of the container C_{t-1} and $\mathcal{D}_{t,q}$ as the corresponding subhypergraph. We break ties by giving higher precedence to lower ranked vertices in the ordering π. We save all the vertices which have higher q-degree than the chosen vertex v_t in any ρn induced subhypergraph induced by vertices in the container C_{t-1} to the set X_q. We then focus on this ρn induced hypergraph to choose the next $(q-1)$ fingerprint vertices. We initialize the fingerprint for the current iteration F^* as F_{t-1} at the start of the inner loop.

For each ℓth (as ℓ goes from $q-1$ to 1) iteration of the inner loop, note that $\mathcal{D}_{t,\ell+1}$ is an $(\ell+1)$-uniform hypergraph with at most $\rho n - (q-(\ell+1))$ vertices. We choose the next fingerprint vertex $v_{t,\ell}$ from $I \setminus F^*$ as the vertex with the maximum $(\ell+1)$-degree in $\mathcal{D}_{t,\ell+1}$ and add it to F^*. Note that we define $\deg_{\mathcal{D}_{t,\ell+1}}(v)$ for a vertex $v \notin \mathcal{D}_{t,\ell+1}$ as zero. We then construct $\mathcal{D}_{t,\ell}$ by removing $v_{t,\ell}$ from the vertex set of $\mathcal{D}_{t,\ell+1}$ and using the edges in $\mathcal{D}_{t,\ell+1}$ incident to $v_{t,\ell}$ minus the vertex $v_{t,\ell}$. We also save the vertices with higher $(\ell+1)$-degree than the current fingerprint vertex $v_{t,\ell}$ in the current hypergraph $\mathcal{D}_{t,\ell+1}$ to the set X_ℓ to be removed later. Note that these vertices cannot be a part of the independent set otherwise they would have been chosen in the fingerprint before the current vertex. We then construct the container set for the next iteration C_t by removing all sets X_ℓ, all vertices with a 1-edge $V[\mathcal{D}_{t,1}]$, and all the new fingerprint vertices $v_{t,\ell}$ from C_{t-1}.

By construction of the algorithm, the fingerprints and containers satisfy

$$F_0 \subseteq F_1 \subseteq F_2 \cdots \subseteq F_{|I|/(q-1)} \subseteq I,$$
$$I \subseteq C_t(F_t) \cup F_t \text{ and } C_{|I|/(q-1)} \subseteq C_{|I|/(q-1)-1} \cdots \subseteq C_0 = V$$

We now prove some important properties of the containers that we use later in our proof to compute an upper bound on the sample complexity of ρ-independent set testing.

▶ **Proposition 2** (Uniqueness of Containers). *For any hypergraph $H = (V, E)$ with a predefined ordering of vertices π, any independent set I in H, and any t, the fingerprint $F_t(I)$ and container $C_t(I)$ of I satisfy*

$$C_t\big(F_t(I)\big) = C_t(I).$$

Proof. For $t \leq \frac{|I|}{q-1}$, the algorithm chooses the same fingerprint vertices in the same order for $F_t(I)$ and I in the first $t-1$ rounds of the algorithm, breaking ties based on the given ordering of vertices π, hence the vertices removed from the container are also identical, i.e., $C_t\big(F_t(I)\big) = C_t(I)$. For $t > \frac{|I|}{q-1}$, $F_t(I) = I$, hence the inequality is satisfied. ◀

We prove another important property of any ℓ-uniform hypergraph which we refer as the standard averaging bound [3].

▶ **Proposition 3** (Standard Averaging Bound). *Let $H = (V, E)$ be an ℓ-uniform hypergraph with $\ell \geq 2$. Then there are at least $\frac{\ell|E|-d|V|}{\binom{|V|-1}{\ell-1}-d} > \frac{\ell|E|-d|V|}{\binom{|V|-1}{\ell-1}}$ vertices in H whose degree is larger than d.*

Algorithm 1 Fingerprint & Container Generator.

Input: A q-uniform hypergraph $H = (V, E)$, a parameter $\rho < 1$, an ordering of vertices π, and an independent set $I \subseteq V$

Initialize $F_0 \leftarrow \emptyset$ and $C_0 \leftarrow V(H)$;

for $t = 1, 2, \ldots, |I|/(q-1)$ **do**

`// Choose` v_t `as the vertex with maximum` q`-degree in any` ρn `induced subhypergraph of` C_{t-1} `and` $\mathcal{D}_{t,q}$ `as the corresponding` ρn `induced subhypergraph. We break ties by giving higher precedence to lower ranked vertices in the ordering` π`.`

$$(v_t, \mathcal{D}_{t,q}) = \underset{v \in I \cap C_{t-1}, D \subseteq C_{t-1}, |D| \leq \rho n}{\arg\max} \deg_D(v) \ ;$$

`// Keep track of all vertices with higher` q`-degree than` v_t `in any` ρn `induced subhypergraph of` C_{t-1} `to remove later`

$X_q \leftarrow \{w \in V[C_t] : \deg^{\leq \rho n}_{C_{t-1}}(w) > \deg^{\leq \rho n}_{C_{t-1}}(v_t)\}$;

`// Initialize the fingerprint for the current iteration`

$F^* \leftarrow F_{t-1}$

for $\ell = q-1, \ldots, 1$ **do**

$v_{t,\ell} \leftarrow$ the vertex in $I \setminus F^*$ with largest $(\ell+1)$ degree in $\mathcal{D}_{t,\ell+1}$;

$F^* \leftarrow F^* \cup v_{t,\ell}$

`// Keep track of all vertices with higher` $(\ell+1)$`-degree than` $v_{t,\ell}$ `in` $\mathcal{D}_{t,\ell+1}$ `to remove later`

$X_\ell \leftarrow \{w \in V[\mathcal{D}_{t,\ell+1}] : \deg_{\mathcal{D}_{t,\ell+1}}(w) > \deg_{\mathcal{D}_{t,\ell+1}}(v_{t,\ell})\}$;

`// Construct` $D_{t,\ell}$ `using vertices of` $D_{t,\ell+1}$ `minus` $v_{t,\ell}$ `and edges incident on` $v_{t,\ell}$

$V[\mathcal{D}_{t,\ell}] \leftarrow V[\mathcal{D}_{t,\ell+1}] \setminus \{v_{t,\ell}\}$;

$E[\mathcal{D}_{t,\ell}] \leftarrow \{e \setminus \{v_{t,\ell}\} : e \in E[\mathcal{D}_{t,\ell+1}] \text{ and } v_{t,\ell} \in e\}$;

end

`// Add` $v_{t,q-1}, \ldots, v_{t,1}$ `to the fingerprint`

$F_t \leftarrow F^*$;

`// Update container set by removing all sets` X_ℓ`, all vertices with a 1-edge, and the new fingerprint vertices`

$C_t \leftarrow C_{t-1} \setminus (\bigcup_{\ell=1}^{q} X_\ell \cup \{v \in C_{t-1} : \{v\} \in E[\mathcal{D}_{t,1}]\} \cup \{v_{t,1}, \ldots, v_{t,q-1}\})$;

end

return $F_1, \ldots, F_{|I|/(q-1)}$ *and* $C_1, \ldots, C_{|I|/(q-1)}$

Proof. Let S be the set of vertices with degree larger than d.

The vertices in S have degree at most $\binom{|V|-1}{\ell-1}$. The remaining $|V| - |S|$ vertices have degree at most d. Since the total number of edges is $1/\ell$ times the sum of degrees, the total number of edges $|E|$ is less than or equal to

$$\frac{1}{\ell}\left(|S|\binom{|V|-1}{\ell-1} + (|V| - |S|)d\right).$$

Rearranging the terms we see that

$$\left(|S|\left[\binom{|V|-1}{\ell-1} - d\right]\right) \geq \ell|E| - d|V|$$

which gives us the desired inequality. ◀

We can now find an upper bound on the maximum q-degree in any ρn-induced subhypergraph of $H[C_t]$.

▶ **Proposition 4** (Maximum Degree Bound). *Let $H = (V, E)$ be a q-uniform hypergraph with $|V| = n$, let I be an independent set in H, and let $t \leq \frac{|I|}{q-1}$. Then the maximum q-degree of every $(\leq \rho n)$-subhypergraph of $H[C_t]$ is at most*

$$\frac{2(q-1)}{\rho t}\binom{\rho n - 1}{q-1}.$$

Proof. Fix any $t \leq \frac{|I|}{q-1}$. Since the containers satisfy $V = C_0 \supseteq C_1 \supseteq \cdots \supseteq C_t$, we have

$$n \geq |C_0| - |C_t| = \sum_{i=1}^{t}\big(|C_{i-1}| - |C_i|\big) = \sum_{i=1}^{t}|C_{i-1} \setminus C_i|.$$

Hence, there exists an index $j \in [t]$ such that $|C_{j-1} \setminus C_j| \leq \frac{n}{t}$.

▷ Claim 5. For each step of the inner loop, $1 \leq \ell \leq q-1$ in the jth iteration, the number of ℓ-edges in $\mathcal{D}_{j,\ell}$ is at most

$$\frac{2\ell}{\rho t}\binom{\rho n - (q-\ell)}{\ell} \tag{1}$$

Proof. We argue by induction on ℓ.

Base case: (ℓ_0 = smallest value of ℓ such that $E[\mathcal{D}_{j,\ell+1}] > 0$)
We observe that $\ell_0 = 1$ or $E[\mathcal{D}_{j,\ell_0}] = 0$ in this case.[3]

(i) $\ell_0 = 1$*:* The vertices corresponding to 1-edges in $\mathcal{D}_{j,1}$ are removed when forming C_j, so their number is at most $|C_{j-1} \setminus C_j| \leq \frac{n}{t}$. Since $q = o(\rho n)$ [4], we have

$$\frac{n}{t} \leq \frac{2}{\rho t}(\rho n - q + 1),$$

so the claim holds for $\ell_0 = 1$.

(ii) $E[\mathcal{D}_{j,\ell_0}] = 0$*:* The claim trivially holds since $E[\mathcal{D}_{j,\ell_0}] = 0 < \frac{2\ell}{\rho t}\binom{\rho n-(q-\ell)}{\ell}$ for any value of ℓ.

Inductive step: Assume Equation (1) holds for $\ell - 1$ where $1 \leq \ell \leq q-2$. Suppose, for contradiction,

$$|E[\mathcal{D}_{j,\ell}]| > \frac{2\ell}{\rho t}\binom{\rho n - (q-\ell)}{\ell}$$

By induction, we have $|E[\mathcal{D}_{j,\ell-1}]| \leq \frac{2(\ell-1)}{\rho t}\binom{\rho n-(q-(\ell-1))}{\ell-1}$, hence $\deg_{\mathcal{D}_{j,\ell-1}}(v_{j,\ell-1}) \leq \frac{2(\ell-1)}{\rho t}\binom{\rho n-(q-\ell)-1}{\ell-1}$.

---

[3] We require $E[\mathcal{D}_{j,\ell_0}] = 0$ as a base case to handle the case that a vertex outside of $\mathcal{D}_{j,\ell+1}$ is added to the fingerprint, because when that happens $|V[\mathcal{D}_{j,\ell}]|$ does not shrink by one, and so the proof in the inductive step does apply as is. As such a vertex is selected then $E[\mathcal{D}_{j,\ell-1}]$ becomes 0.

[4] We observe that the sample complexity bound of Theorem 1 holds trivially when $q = \Omega(\rho n)$. Notice that the number of vertices is $O\left(\frac{q}{\rho}\right)$ in this case so the sample complexity is $O\left(\frac{q}{\rho}\right) < O\left(\frac{1}{\varepsilon}\right)$. We remind the reader that the problem is non-trivial only when $\varepsilon < \frac{\rho^q}{q!}$. We can therefore assume $q = o(\rho n)$ for the remainder of this paper.

Therefore, by the standard averaging bound (Proposition 3) with $d = (\ell-1)\frac{|E|}{|V|}$,

$$\frac{|E[\mathcal{D}_{j,\ell}]|}{\binom{|V[\mathcal{D}_{j,\ell}]|-1}{\ell-1}} > \frac{\frac{2\ell}{\rho t}\binom{\rho n-(q-\ell)}{\ell}}{\binom{\rho n-(q-\ell)-1}{\ell-1}} = \frac{2}{\rho t}\big(\rho n-(q-\ell)\big) > \frac{n}{t}$$

where we have used the fact that $|V[\mathcal{D}_{j,\ell}]| \le \rho n-(q-\ell)$.

Hence there are more than n/t vertices in some ρn-induced subhypergraph of $\mathcal{D}_{j,\ell}$ with degree exceeding

$$(\ell-1)\,\frac{|E[\mathcal{D}_{j,\ell}]|}{|V[\mathcal{D}_{j,\ell}]|} > (\ell-1)\cdot\frac{\frac{2\ell}{\rho t}\binom{\rho n-(q-\ell)}{\ell}}{\rho n-(q-\ell)} = \frac{2(\ell-1)}{\rho t}\binom{\rho n-(q-\ell)-1}{\ell-1}$$

All such vertices are removed from C_j (since they have higher degree than $v_{j,\ell}$), implying $|C_{j-1}\setminus C_j| > n/t$, a contradiction. This completes the induction and the proof of the claim. ◁

Why the claim implies the proposition: We consider the jth iteration of the algorithm. Taking $\ell = q-1$, the number of $(q-1)$-edges in $\mathcal{D}_{j,q-1}$ is at most $\frac{2(q-1)}{\rho t}\binom{\rho n-1}{q-1}$. Therefore, the fingerprint vertex $v_{j,q-1}$ or v_j has q-degree at most $\frac{2(q-1)}{\rho t}\binom{\rho n-1}{q-1}$ in any $(\le \rho n)$-induced subgraph of $H[C_{j-1}]$. All vertices exceeding this degree in some $(\le \rho n)$-subgraph of $H[C_{j-1}]$ are excluded from C_j, so the maximum q-degree in any $(\le \rho n)$-subgraph of $H[C_j]$ is at most $\frac{2(q-1)}{\rho t}\binom{\rho n-1}{q-1}$. Since $C_t \subseteq C_j$, the same bound holds for every $(\le \rho n)$-subgraph of $H[C_t]$. ◀

2.2 Proof of the New Container Lemma

We can now prove the new container lemma which essentially states that the container size is "small" (as defined in the lemma) for some small value of t independent of the hypergraph size n when the input hypergraph is ε-far from ρn independent set. We prove the contrapositive, i.e., we construct a ρn-induced subhypergraph which is not ε-far from ρ-independent set property if the container size is "large" for all values of t.

▶ **Lemma 6** (New Container Lemma). *Let $H=(V,E)$ be a q-uniform hypergraph on n vertices that is ε-far from ρ-independent set property. Then, for any independent set I in H, there exists an index $t \le \frac{8\rho^{q-1}}{\varepsilon(q-1)!}$ such that the size of the tth container of I satisfies*

$$|C_t| \le \left(\rho - t\cdot\frac{\varepsilon(q-2)!}{4\rho^{q-2}\ln(\rho^{q-1}/\left(\varepsilon(q-1)!\right))}\right)n$$

Proof. We prove the contrapositive. Let $t_1 = \frac{8\rho^{q-1}}{\varepsilon(q-1)!}$. Suppose that for all $t \le t_1$, the size of the tth container $C_t(I)$ satisfies

$$|C_t(I)| > \left(\rho - t\cdot\frac{\varepsilon(q-2)!}{4\rho^{q-2}\ln(\rho^{q-1}/(\varepsilon(q-1)!))}\right)n \tag{2}$$

We will show that there is some ρn induced subhypergraph with $< \varepsilon n^q$ edges showing that H is not ε-far from ρ-independent set property.

We start with the container $C_{t_1}(I)$ to construct the set of vertices S and the corresponding induced subhypergraph $H[S]$. Add vertices one at a time from the previous containers $C_t(I)$ (in decreasing order of t) until the size of the set reaches ρn. Note that we can choose

any set of ρn vertices from $C_{t_1}(I)$ if the size of the container is greater than ρn. Let V_t denote the set of vertices in container $(t-1)$ but not in container t in this set S, i.e., $V_t = \{v : v \in S \text{ and } v \in C_{t-1} \backslash C_t\}$ and let t_0 be the smallest value of t for which $|V_t| > 0$.

We now prove the claim. Since $|S| \leq \rho n$, Proposition 4 implies that the maximum q-degree in any ρn induced subhypergraph of $H[C_{t_1}(I)]$ is at most

$$\frac{\frac{2(q-1)}{\rho}\binom{\rho n-1}{q-1}}{\frac{8\rho^{q-1}}{\varepsilon(q-1)!}} \leq \frac{\varepsilon q!}{4\rho^q}\binom{\rho n-1}{q-1}$$

Hence, any $(\leq \rho n)$-induced subhypergraph of $H[C_{t_1}(I)]$ has at most $\frac{\varepsilon q!}{4\rho^q}\binom{\rho n}{q}$ q-edges.

When $|C_{t_1}| \geq \rho n$, then the total number of edges is at most

$$\frac{\varepsilon q!}{4\rho^q}\binom{\rho n}{q} < \frac{\varepsilon q!}{\rho^q}\frac{(\rho n)^q}{q!} < \varepsilon n^q,$$

therefore $H[S]$ is not ε-far from ρn independent set.

Now we consider the case when $|C_{t_1}| < \rho n$. By Proposition 4, each $v \in V_t$ contributes at most

$$\frac{2(q-1)}{\rho(t-1)}\binom{\rho n-1}{q-1}$$

additional edges (or $\binom{\rho n-1}{q-1}$ if $t=1$), since v lies in $C_{t-1}(I)$.

Therefore, the total number of edges is at most

$$\frac{\varepsilon q!}{4\rho^q}\binom{\rho n}{q} + |V_1| \cdot \binom{\rho n-1}{q-1} + \sum_{t=2}^{t_1} |V_t| \cdot \frac{2(q-1)}{\rho(t-1)}\binom{\rho n-1}{q-1} \tag{3}$$

For any $t_0 \leq t \leq t_1$, the set $\bigcup_{j=1}^{t} V_j$ corresponds to the vertices not appearing in $C_t(I)$, and so Equation (2) implies

$$\sum_{j=1}^{t} |V_j| \leq \rho n - |C_t| < \frac{t\varepsilon n(q-2)!}{4\rho^{q-2}\ln(\rho^{q-1}/(\varepsilon(q-1)!))}$$

In the sum (3), the contribution from $|V_t|$ decreases as t increases, so the sum is maximized when all $|V_t| = \frac{\varepsilon n(q-2)!}{4\rho^{q-2}\ln(\rho^{q-1}/\varepsilon(q-1)!)}$. Substituting into (3), we obtain

$$\begin{aligned}
&\frac{\varepsilon q!}{4\rho^q}\binom{\rho n}{q} + \frac{\varepsilon n(q-2)!}{4\rho^{q-2}\ln(\rho^{q-1}/(\varepsilon(q-1)!))}\binom{\rho n-1}{q-1} \\
&\qquad + \frac{\varepsilon n(q-2)!}{4\rho^{q-2}\ln(\rho^{q-1}/(\varepsilon(q-1)!))} \sum_{t=2}^{\frac{8\rho^{q-1}}{\varepsilon(q-1)!}} \frac{2(q-1)}{\rho(t-1)}\binom{\rho n-1}{q-1} \\
&< \frac{\varepsilon q!}{4\rho^q}\binom{\rho n}{q} + \frac{\varepsilon q!}{4(q-1)\rho^{q-1}\ln(\rho^{q-1}/(\varepsilon(q-1)!))}\binom{\rho n}{q} \\
&\qquad + \frac{2\varepsilon q!}{4\rho^q\ln(\rho^{q-1}/(\varepsilon(q-1)!))} \sum_{t=2}^{\frac{8\rho^{q-1}}{\varepsilon(q-1)!}} \frac{1}{(t-1)}\binom{\rho n}{q} \\
&< \frac{\varepsilon q!}{\rho^q}\binom{\rho n}{q}\left(\frac{1}{4} + \frac{\rho}{4(q-1)\ln(\rho^{q-1}/(\varepsilon(q-1)!))} + \frac{1}{2\ln(\rho^{q-1}/(\varepsilon(q-1)!))} \sum_{t=2}^{\frac{8\rho^{q-1}}{\varepsilon(q-1)!}} \frac{1}{t-1}\right) \\
&< \frac{\varepsilon q!}{\rho^q}\binom{\rho n}{q} < \frac{\varepsilon}{\rho^q}(\rho n)^q < \varepsilon n^q,
\end{aligned}$$

where the final inequality uses the upper bound $H(m) \leq \ln(m) + 1$ on the harmonic series and the assumption that ε is less than a sufficiently large constant, i.e., $\varepsilon < e^{-8}\frac{\rho^{q-1}}{(q-1)!}$ (which can be made without loss of generality as we can always choose a smaller value of ε' for the tester such that $e^{-8}\varepsilon < \varepsilon' < \varepsilon$ which gives us the same sample complexity). Note that $\varepsilon < \frac{\rho^q}{q!} < \frac{\rho^{q-1}}{(q-1)!}$ since the graph is ε-far from ρ-independent set property.

This completes the proof. ◀

▶ Remark 7. There exists a q-uniform hypergraph H that is ε-far from having a ρn independent set and an independent set I for which Lemma 6 is nearly tight in the ρ-dependence. Consider the same example from the lower bound of Langberg [8], and re-analyzed in Appendix A, which is a hypergraph H formed by planting an independent set I of size $\left(\rho - \frac{4(q-1)!\varepsilon}{\rho^{q-1}}\right)n$ in a complete q-uniform hypergraph. See Appendix A for a justification that this hypergraph is ε-far from having a ρn independent set.

We claim that for all t we have that $|C_t(I)| \geq \left(\rho - \frac{16(q-1)!\varepsilon t}{\rho^{q-2}}\right)n$, which demonstrates that Lemma 6 is roughly tight in the ρ dependence. To show this, first consider the case that $t \leq \frac{1}{2\rho}$. Observe that for the first $\frac{1}{2\rho}$ iterations of the outer loop of the container procedure, exactly ρn vertices are removed from the container in each iteration. This is because no vertices will be removed due to having larger degree in a ρn subhypergraph (every vertex has full degree in some ρn subhypergraph). Hence, $|C_t(I)| \geq \rho n$ for all $t \leq \frac{1}{2\rho}$.

Next consider the case that $t > \frac{1}{2\rho}$. Recall that at every step t of the container procedure we have that $I \subseteq C_t(I) \cup F_t(I)$, and so $|C_t(I)| \geq |I \setminus F_t(I)| \geq \left(\rho - \frac{8(q-1)!\varepsilon}{\rho^{q-1}}\right)n$, where we are assuming that $|F_t(I)|$ is sufficiently small. Hence, for $t > \frac{1}{2\rho}$, we find that $|C_t(I)| \geq \left(\rho - \frac{16(q-1)!\varepsilon t}{\rho^{q-2}}\right)n$, completing the proof of the claim.

▶ Remark 8. It is not possible to achieve our bound in Lemma 6 and Theorem 1 using a standard hypergraph container procedure. In particular, a standard hypergraph container procedure run on a q-uniform hypergraph $H = (V, E)$ and independent set I would initialize a container to V and run in iterations. In each iteration it would run an inner loop that selects a vertex v in the independent set with highest degree in the hypergraph on the current container, then look at the $(q-1)$-uniform hypergraph formed by all edges incident to v, and so on until reaching the 1-uniform hypergraph. This is in stark contrast to the algorithm described in Section 2.1 that selects vertices with highest degree in some ρn subhypergraph. We argue that this natural hypergraph procedure is not useful for proving our bounds via an example described below.

Similar to the the lower bound of Langberg [8] and re-analyzed in Appendix A, start by planting a large independent set I of size $\left(\rho - \frac{8(q-1)!\varepsilon}{\rho^{q-1}}\right)n$ in an n-vertex hypergraph. Between I and $V \setminus I$ add all possible edges with $q-1$ endpoints in I and 1 endpoint in $V \setminus I$. The hypergraph $H[V \setminus I]$ is going to be a random hypergraph with edge density $\rho^{q-2} \cdot (q-1)$ so that the vertices in $V \setminus I$ have degree less than roughly $\rho^{q-2}(q-1) \cdot \binom{n}{q} \cdot \frac{q}{n} \leq \frac{\rho^{q-2}n^{q-1}}{(q-2)!}$, which is roughly the degree of vertices in I. Finally, select a small subset $I' \subset I$ of size roughly q/ρ^{q-2} and remove all edges in the hypergraph with at least one endpoint in I'. Instead, for each $v \in I'$ select $2\frac{\rho^{q-2}}{(q-2)!}n^{q-1}$ random $(q-1)$ tuples from $V \setminus v$ and add an edge containing v and those $q-1$ vertices. Crucially we want the vertices in I' to have highest degree in the hypergraph, but whose degree is similar to the vertices in I.

It is not hard to show that H is ε-far from having a ρn independent set with high probability (over the randomness of the edges added in the hypergraph), assuming that ε is sufficiently small compared to ρ. Further, we claim that running the standard hypergraph

container procedure above would take at least roughly $\frac{1}{\rho^{q-2}}$ steps to get a desirable container. This is because the vertices in I' have highest degree and so they are selected in each step of the procedure until they have all been selected. Since the edges of vertices in I' are randomly placed throughout the hypergraph with density roughly $2(q-1)\rho^{q-2}$, the standard hypergraph container procedure removes roughly $2(q-1)\rho^{q-2}n$ vertices in each step of the outer loop. Hence it will take at least roughly $1/\rho^{q-2}$ steps until $|C_t(I)| < \rho n$.

Comparing with our hypergraph procedure given in Section 2.1 on this hypergraph construction, since at every step t of the container procedure we have that $I \subseteq C_t(I) \cup F_t(I)$, then $|C_t(I)| \gtrsim \left(\rho - \frac{8(q-1)!\varepsilon}{\rho^{q-1}}\right) n$. Hence, the t given in Lemma 6 is at most $\frac{32(q-1)\ln(1/\varepsilon)}{\rho}$, and so the number of steps required by a standard hypergraph is larger than the number of steps given in Lemma 6 by a factor of at least $1/\rho^{\Theta(q)}$.

3 Upper Bound on Sample Complexity

The proof of the theorem uses the following form of Chernoff's bound for hypergeometric distributions [4].

▶ **Lemma 9** (Chernoff's bound). *Let X be drawn from the hypergeometric distribution $H(N, K, m)$, where m elements are drawn without replacement from a population of N elements, K of which are marked. Let X denote the number of marked elements drawn. Then for any $\vartheta \geq \mathbb{E}[X]$,*

$$\Pr\left[X \geq \vartheta\right] \leq \exp\left(-\frac{(\vartheta - \mathbb{E}[X])^2}{\vartheta + \mathbb{E}[X]}\right).$$

▶ **Theorem 1.** *The sample complexity for testing the ρ-independent set property is $\widetilde{O}\left(\frac{q\rho^{2q-3}}{\varepsilon^2(q-2)!^2}\right)$.*

Proof. Let S be a random set of $s = c \cdot \frac{q\rho^{2q-3}}{\varepsilon^2(q-2)!^2} \ln^3\left(\frac{\rho^{q-1}}{\varepsilon(q-1)!}\right)$ vertices drawn uniformly at random from V without replacement, where c is a large enough constant. We ignore the integer rounding issues in the rest of the proof as they do not affect the asymptotics of the final result. The ε-tester checks if the induced subhypergraph $H[S]$ contains an independent size of ρs, if yes, the tester accepts and rejects otherwise.

If G contains a ρn independent set, then S contains at least ρs vertices from this independent set with probability at least $1/2$, since the number of such vertices follows a hypergeometric distribution, and the median of this distribution is at least ρs.

In the remainder of the proof we upper bound the probability that S contains a ρs-independent set when G is ε-far from containing a ρn-independent set. For the rest of the argument, let us call a container C_t small when its cardinality is bounded by $|C_t| \leq \left(\rho - t \cdot \frac{\varepsilon(q-2)!}{4\rho^{q-2}\ln(\rho^{q-1}/(\varepsilon(q-1)!))}\right) n$, the expression in the lemma above.

Let us denote the vertices of the sampled set S as $u_1, u_2, \ldots, u_s$. First observe that, by Lemma 6, for every independent set I of size ρs in S, there is a $t \leq \frac{8\rho^{q-1}}{\varepsilon(q-1)!}$ and a fingerprint $F_t \subseteq I$ of size $(q-1)t$ that defines a small container $C_t(F_t)$ such that $I \subseteq C_t(F_t) \cup F_t$. Hence, the probability that S contains an independent set of size at least ρs is at most the probability that there exists some $t \leq \frac{8\rho^{q-1}}{\varepsilon(q-1)!}$ such that S contains a set of $t' = (q-1)t$ vertices that form the fingerprint F_t of a small container C_t, and S contains at least $\rho s - t'$ other vertices from C_t. We now upper bound this probability.

For any t' distinct indices $i_1, i_2, \ldots, i_{t'} \in [s]$, consider the event where the vertices $u_{i_1}, \ldots, u_{i_{t'}}$ form the fingerprint F_t of a small container C_t. Let X denote the number of vertices among the other $s - t'$ sampled vertices that belong to C_t. By the bound on the size of small containers, the expected value of X is

$$\begin{aligned} E[X] &\leq \left(\rho - t \cdot \frac{\varepsilon(q-2)!}{4\rho^{q-2}\ln(\rho^{q-1}/(\varepsilon(q-1)!))}\right)(s-t') \\ &< \rho s - \frac{t\varepsilon s(q-2)!}{4\rho^{q-2}\ln(\rho^{q-1}/(\varepsilon(q-1)!))} \\ &\leq \rho s - t' - \frac{t\varepsilon s(q-2)!}{8\rho^{q-2}\ln(\rho^{q-1}/(\varepsilon(q-1)!))}, \end{aligned}$$

where the last inequality uses the fact that $s = c \cdot \frac{q\rho^{2q-3}}{\varepsilon^2(q-2)!^2}\ln^3\left(\frac{\rho^{q-1}}{\varepsilon(q-1)!}\right)$ for a large enough constant c and that the problem is non-trivial only when $\varepsilon < \frac{\rho^q}{q!}$.

By the Chernoff bound (Lemma 9), the probability that we draw at least $\rho s - t'$ vertices from C_t in the final $s - t'$ vertices drawn to form S is

$$\begin{aligned} \Pr[X \geq \rho s - t'] &\leq \exp\left(-\frac{(\rho s - t' - E[X])^2}{\rho s - t' + E[X]}\right) \\ &\leq \exp\left(-\frac{1}{2\rho s}\frac{t^2\varepsilon^2 s^2(q-2)!^2}{64\rho^{2q-4}\ln^2(\rho^{q-1}/(\varepsilon(q-1)!))}\right) \\ &\leq \exp\left(-\frac{t^2\varepsilon^2(q-2)!^2 s}{128\rho^{2q-3}\ln^2(\rho^{q-1}/(\varepsilon(q-1)!))}\right). \end{aligned}$$

Therefore, by applying a union bound over all possible choices of t' indices from $[s]$, the probability that any set of at most $\frac{8\rho^{q-1}}{\varepsilon(q-2)!}$ vertices in S form the fingerprint of a small container from which we sample at least $\rho s - t'$ vertices in S is at most

$$\begin{aligned} &\sum_{t'}\binom{s}{t'}\exp\left(-\frac{t'^2\varepsilon^2 s}{128(q-1)^2\rho^{2q-3}\ln^2(\rho^{q-1}/(\varepsilon(q-1)!))}\right) \\ &\leq \sum_{t}\exp\left((q-1)t\ln s - \frac{t^2\varepsilon^2(q-2)!^2 s}{128\rho^{2q-3}\ln^2(\rho^{q-1}/(\varepsilon(q-1)!))}\right) \\ &< \frac{1}{3} \end{aligned}$$

where the first inequality uses the upper bound $\binom{s}{t'} \leq s^{t'}$, and the last inequality again uses the fact that $s = c \cdot \frac{q\rho^{2q-3}}{\varepsilon^2(q-2)!^2}\ln^3\left(\frac{\rho^{q-1}}{\varepsilon(q-1)!}\right)$ for a large enough constant c. Hence, the probability that S contains an independent set of size at least ρs is less than $1/3$. ◀

References

1. Noga Alon and Asaf Shapira. A characterization of the (natural) graph properties testable with one-sided error. *SIAM Journal on Computing*, 37(6):1703–1727, 2008. doi:10.1137/06064888X.
2. József Balogh, Robert Morris, and Wojciech Samotij. Independent sets in hypergraphs. *Journal of the American Mathematical Society*, 28(3):669–709, 2015.
3. Eric Blais and Cameron Seth. New graph and hypergraph container lemmas with applications in property testing. In *Proceedings of the 56th Annual ACM SIGACT Symposium on Theory of Computing*, 2024.

4 Eric Blais and Cameron Seth. Testing graph properties with the container method. *SIAM Journal on Computing*, 54(6):1626–1642, 2025. doi:10.1137/24M1653173.

5 Uriel Feige, Michael Langberg, and Gideon Schechtman. Graphs with tiny vector chromatic numbers and huge chromatic numbers. *SIAM Journal on Computing*, 33(6):1338–1368, 2004. doi:10.1137/S0097539703431391.

6 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998. doi:10.1145/285055.285060.

7 Daniel J. Kleitman and Kenneth J. Winston. On the number of graphs without 4-cycles. *Discrete Mathematics*, 41(2):167–172, 1982. doi:10.1016/0012-365X(82)90204-7.

8 Michael Langberg. Testing the independence number of hypergraphs. In *Approximation, Randomization, and Combinatorial Optimization, Algorithms and Techniques, 7th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2004, and 8th International Workshop on Randomization and Computation, RANDOM 2004, Cambridge, MA, USA, August 22-24, 2004, Proceedings*, pages 405–416, 2004. doi:10.1007/978-3-540-27821-4_36.

9 Alexander Sapozhenko. Systems of containers and enumeration problems. In *Stochastic Algorithms: Foundations and Applications (SAGA 2005)*, pages 1–13. Springer, 2005. doi:10.1007/11571155_1.

10 David Saxton and Andrew Thomason. Hypergraph containers. *Inventiones mathematicae*, 201(3):925–992, 2015.

A Lower Bound on Sample Complexity

We prove a lower bound for the sample complexity of ρ-independent set testing using the construction by Langberg [8].

▶ **Proposition 10.** *Let n be a sufficiently large constant. Let $\rho > 0$ and $\varepsilon > 0$ satisfy the following conditions:*

- $\varepsilon \ll \dfrac{\rho^q}{2q!}$,
- $\dfrac{\rho^{2q-1}}{\varepsilon^2} \ll n$,
- $q^2 \ll \rho n$,

Then there exists a hypergraph H on n vertices such that H is ε-far from ρ-independent set property, and with probability at least $e^{-1/2}$, a random subhypergraph H of size

$$s = \frac{\rho^{2q-1}}{8\left((q-1)!\right)^2 \varepsilon^2}$$

contains an independent set of size ρs.

Proof. Consider the q-uniform hypergraph $H = (V, E)$ defined as follows:

i) $|V| = n$,

ii) V is partitioned into two disjoint sets A and $V \setminus A$, where

$$|A| = \left(1 - \frac{4(q-1)!\varepsilon}{\rho^q}\right)\rho n,$$

iii) the edge set E consists of all q-subsets of V except those fully contained in A (i.e., A is an independent set).

We first show that H is ε-far from ρ-independent set property, i.e., every subset of vertices of size ρn induces a subhypergraph with at least εn^q edges.

Consider any subset $S \subseteq V$ of size ρn. We define $\delta := \frac{4(q-1)!\varepsilon}{\rho^q}$ so $|A| = (1-\delta)\rho n$.

Thus,

$$\begin{aligned}
|E(S)| &\geq \binom{\rho n}{q} - \binom{(1-\delta)\,\rho n}{q} \\
&\geq \frac{(\rho n - q)^q}{q!} - \frac{((1-\delta)\rho n)^q}{q!} \\
&\geq \frac{(\rho n)^q}{q!}\left[\left(1 - \frac{q}{\rho n}\right)^q - (1-\delta)^q\right] \\
&\geq \frac{(\rho n)^q}{q!}\left[\left(1 - \frac{q^2}{\rho n}\right) - \left(1 - \frac{\delta q}{2}\right)\right] \qquad (*) \\
&\geq \frac{(\rho n)^q}{q!}\left[\left(\frac{\delta q}{2} - \frac{q^2}{\rho n}\right)\right] \\
&\geq \frac{(\rho n)^q}{q!}\frac{\delta q}{4}
\end{aligned}$$

where (*) follows from $(1-a)^x \geq 1 - ax$ and $(1-a)^x \leq e^{-ax} \leq (1 - \frac{ax}{2})$ when $ax < 1$ using Taylor's expansions and the last inequality follows when $q < \frac{\delta \rho n}{4}$.

Substituting $\delta = \frac{4(q-1)!\varepsilon}{\rho^q}$ gives $|E(S)| \geq \varepsilon n^q$.

Now we consider a sample of $s = \frac{\rho^{2q-1}}{8((q-1)!)^2\varepsilon^2}$ vertices chosen randomly from the hypergraph H. We define $X := |S \cap A| = \sum_{v \in A} \mathbf{1}_{\{v \in S\}}$. Then X follows a hypergeometric distribution with parameters $(n, |A|, s)$, and mean $\mu = \frac{|A|}{n} \cdot s = \left(1 - \frac{4(q-1)!\varepsilon}{\rho^q}\right)\rho s$.

Thus, using the Central Limit Theorem, we obtain that for sufficiently large n,

$$\Pr\left[|S \cap A| > \left(1 - \frac{4(q-1)!\varepsilon}{\rho^q}\right)\rho s + \sqrt{\rho s} = \rho s\right] \geq \Pr[N(0,1) > 3/2] \geq \frac{1}{20}$$

Therefore, the chosen sample has an independent set of size ρs with some constant positive probability and the sample size $s = \frac{\rho^{2q-1}}{8((q-1)!)^2\varepsilon^2}$ gives a lower bound for testing ρn independent set. ◀

Toward a KKL Theorem for Any HDX

Max Hopkins
Institute for Advanced Study, Princeton, NJ, USA

Abstract

The KKL Theorem, a seminal result in boolean function analysis, characterizes the structure of low-influence (non-expanding) functions on the hypercube. While recent years have seen breakthrough results across a variety of areas relying on analogs of the KKL Theorem *beyond the cube* (e.g. on product spaces, Grassmann graphs), further progress has been inhibited by our poor understanding of the phenomenon across more general domains. Motivated in this context, Bafna, Hopkins, Kaufman, and Lovett (STOC 2022) and Gur, Lifshitz, and Liu (STOC 2022) proved a generalized KKL-type Theorem for spectral high dimensional expanders (HDX). Their results, however, remain highly restricted due to strong quantitative expansion requirements on the underlying complex.

In this work, we introduce a simple local-to-global method for analyzing low influence functions on simplicial complexes. Using this method we prove a local-to-global KKL-type Theorem: any simplicial complex whose links satisfy a KKL-Theorem also satisfies such a result globally. Building on Gotlib and Kaufman (RANDOM 2023), we also prove a weaker dimension-dependent KKL-type Theorem for simplicial complexes with *any non-trivial (two-sided) expansion*. As concrete applications of our framework, we give the first characterization of non-expanding functions on "combinatorial" HDX such as dense clique complexes and a corresponding Kruskal-Katona Theorem, as well as a small-set expansion theorem for the Ramanujan Complexes of Lubotzky, Samuels, and Vishne (EJC '05).

2012 ACM Subject Classification Theory of computation

Keywords and phrases High Dimensional Expanders, Boolean Function Analysis

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.74

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2606.29449`

Funding *Max Hopkins*: Supported by NSF Award DMS-2424441.

1 Introduction

In 1988, Kahn, Kalai, and Linial [51] proved the KKL Theorem: a characterization of low influence (non-expanding) functions[1] on the hypercube that revolutionized the field of boolean function analysis. In recent years, variants of the KKL Theorem *beyond the cube*, including on products [13, 70, 38, 36, 46, 56, 63], Lie groups [30], and the Grassmann [62, 29] have become a powerful tool across a surprisingly broad array of areas, leading, e.g., to the theory of sharp thresholds [37], multiple breakthroughs in PCP theory [62, 61, 27, 26, 10, 60, 66], new streaming lower bounds [32], and to progress on longstanding problems in group theory and combinatorics [30, 58, 57, 59, 43]. Despite its broad importance our understanding of KKL-type theorems on general spaces remains poor, standing as a barrier to further progress on key problems such as the unique games conjecture [10].

Toward this end, a series of works over the past several years [55, 19, 6, 7, 44, 39, 47] developed a new framework for unifying our understanding of Fourier analysis and KKL-type Theorems beyond the cube: *spectral high dimensional expanders* (HDX). HDX are

[1] Informally, a low-influence function on the hypercube is one which has low expected probability of changing value under a random coordinate flip.

© Max Hopkins;
licensed under Creative Commons License CC-BY 4.0

a generalization of expanders to hypergraphs and ranked posets that have seen a recent explosion of application within theoretical computer science [55, 19, 6, 7, 44, 39, 25, 18, 53, 2, 24, 48, 11, 5, 3, 4, 16, 15, 14, 33, 50, 64, 12, 23, 68, 28, 21, 20, 8, 9, 49, 45, 22]. In particular, Bafna, Hopkins, Kaufman, and Lovett [6, 7], Gur, Lifshitz, and Liu [44], and Hopkins [47] proved a KKL-Type Theorem for certain spectral high dimensional expanders stating that any low influence function must be *local*, highly concentrated in some restriction of the hypergraph. Unfortunately, these results only hold for the regime of *near-perfect* expansion, a strong requirement satisfied only by a few families of sparse hypergraphs [65, 54, 17].

In this work, we develop simple local-to-global machinery for analyzing low influence functions on hypergraphs allowing us to remove these requirements in certain well-studied settings. We prove two main results: a KKL-Type Theorem for simplicial complexes whose links (local restrictions) themselves satisfy such a bound such as quotients of the affine building [65] (the so-called Ramanujan complexes), and a weaker (dimension-dependent) KKL-type Theorem for simplicial complexes with *any* non-trivial (two-sided) spectral high dimensional expansion allowing us to give the first non-trivial characterization of low influence functions on product-based HDX [40] and dense clique complexes. We also derive Kruskal-Katona theorems for these objects, a broadly used set of results in extremal combinatorics bounding the lower shadow of a set system.

Our work builds on a growing body of Fourier analytic tools for high dimensional expanders [55, 19, 6, 7, 44, 39, 18, 2, 42] and relies in particular on a decomposition theorem of Gotlib and Kaufman (GK) [42]. We make two main technical contributions in this line. First, we show for any hypergraph X whose *negative* local spectra is bounded away from -1, "global" functions on X (i.e. functions not concentrated in any local component) have bounded projection onto low levels of the GK-decomposition. Second, we consider an *inductive* approach for bounding expansion through a new "function-dependent" variant of Garland's Lemma, a way to write the expansion of f as an expectation across local parts of the complex where, unlike prior such methods, the underlying distribution may depend on the function f in question. In a similar vein, our local-to-global theorem for (global) small-set expansion combines this latter Garland method with Fourier analytic properties of a localized noise operator to derive global expansion bounds.

2 Background

2.1 High Dimensional Expanders

We focus on weighted *pure simplicial complexes* (X, Π) where

$$X = X(0) \cup \ldots \cup X(d)$$

for $X(d) \subset \binom{n}{d}$ an arbitrary d-uniform hypergraph and $X(i) \subset \binom{n}{i}$ given by downward closure, and

$$\Pi = (\pi_0, \ldots, \pi_d)$$

for π_d an arbitrary distribution over $X(d)$ and π_i given by selecting a uniformly random i-set from $\tau \sim \pi_d$.

Our bounds will depend on the now standard notion of high dimensional expansion which bounds the behavior of local neighborhoods of X called *links*. For every face $\tau \in X(i)$, the link of τ is given by:

$$X_\tau \coloneqq \{\sigma \setminus \tau \in X : \tau \subseteq \sigma \in X\}$$

Given a function $f : X(k) \to \mathbb{R}$, its localization $f|_\tau : X_\tau(k - |\tau|) \to \mathbb{R}$ is $f_\tau(\sigma) = f(\tau \cup \sigma)$.

Denote the normalized adjacency matrix of the graph underlying X_τ by A_τ, and for $0 \le j \le d-2$ denote the worst-case spectral parameters of j-links by

$$\gamma_j \coloneqq \max_{\tau \in X(j)} \{\lambda_2(A_\tau)\}, \qquad \gamma_j^{(-)} \coloneqq \min_{\tau \in X(j)} \{\lambda_{\min}(A_\tau)\}.$$

A complex is called *strongly connected* if every $\gamma_j < 1$, a (one-sided) *local-spectral expander* if every γ_j is bounded away from 1 [52, 67, 25], and a (two-sided) local-spectral expander if every $\gamma_j^{(-)}$ is additionally bounded away from -1.

2.2 Influence and the Down-Up Walk

As in the seminal work of KKL, we are interested in characterizing the structure of boolean functions $C_k(X, \mathbb{F}_2) \coloneqq \{f : X(k) \to \{0,1\}\}$ with low total influence. In our setting it will be convenient to use an equivalent definition (up to normalization) concerning the expansion of a celebrated random walk on complexes called the *lower* or *down-up* walk. The lower walk at level k, denoted N_k^1, moves between k-faces of the complex via a shared $(k-1)$-face, removing a vertex uniformly at random and re-sampling from the appropriate conditional distribution on the remaining $(k-1)$-face. The *edge expansion* of a set $S \subset X(k)$ with respect to the lower walk is the expected probability of leaving S in a single step, or equivalently:

$$\Phi(S) = 1 - \frac{\langle N_k^1 1_S, 1_S \rangle}{\langle 1_S, 1_S \rangle}.$$

It is not hard to see that expansion, which can also be written as $\frac{\langle 1_S, L 1_S \rangle}{\langle 1_S, 1_S \rangle}$ for the "Laplacian Operator" L, is equivalent to standard notions of influence up to normalization (see full version for details), so we will focus on this notion throughout instead.

2.3 Small-Set Expansion and the Noise Operator

The small-set expansion theorem is a classical and closely related result to the KKL theorem [1, 51]. In its standard form, it states that small sets on the noisy cube expand near perfectly. To prove an analog result beyond the cube, we need to define a corresponding notion of the noise operator T_ρ on simplicial complexes. A natural generalization is to consider the walk on k-faces of X which fixes each vertex with probability ρ, and re-samples the remaining vertices. Formally, we may express this operator as a convex combination of "longer" down-up walks

$$T_\rho \coloneqq \sum_{i=0}^{k} \binom{k}{i} (1-\rho)^{k-i} \rho^i N_k^{k-i},$$

where N_k^j moves between k-faces via a shared $(k-j)$-face. Note this recovers the standard operator when X is a product space (see [7]). We will always write expansion with respect to the noise operator at Φ_{T_ρ}, and write just Φ for the down-up walk. Finally, it will sometimes be useful to reference the opposite variant of the lower walks, the "upper" walks $\widehat{N}_k^i$, which moves between k-faces via a shared $(k+i)$-face.

3 Results

We are now ready to more formally cover our results. We split the subsection into two parts: structure theorems for low-influence functions on arbitrary HDX, and a local-to-global small-set expansion theorem.

3.1 A Structure Theorem for Low Influence Functions

The canonical examples of low influence (non-expanding) sets on simplicial complexes are *links*.[2] Our goal is to show these are the *only* such examples. To this end, we call a function f (ε, i)-global if it is uncorrelated with i-links:

$$\forall \tau \in X(i) : \mathop{\mathbb{E}}_{X_\tau}[f_\tau] \leq \mathbb{E}[f] + \varepsilon.$$

We prove that global functions indeed expand, where the quantitative parameters depend on the underlying local-spectral expansion of the complex.

▶ **Theorem 1** (Expansion of Global Functions). *Let (X, Π) be a k-dimensional simplicial complex and $f : X(k) \to \mathbb{R}$ any (ε, i)-global boolean function. Then*

$$\Phi(f) \geq \frac{1 - \mathbb{E}[f]}{k - i} \prod_{j=i}^{k-2} (1 - \gamma_j) - c_{k,i,\gamma}\varepsilon.$$

where $c_{k,i,\gamma} \leq (k - i + 1)\left(\frac{1}{k-i}\prod_{j=i}^{k-2}(1-\gamma_j) - \frac{1}{k-i+1}\prod_{j=i-1}^{k-2}(1-\gamma_j)\right)\left(1 + (k-i)\gamma_{i-1}^{(-)}\right)^{-1}$

For many regimes of interest, e.g. when either γ_{k-2} or $|\gamma_{k-2}^{(-)}|$ is less than roughly $\frac{1}{k}$, $c_{\gamma,k,i} \leq O(1)$ is an *absolute constant*, with no dependence on dimension. In such cases the contrapositive of Theorem 1 states *any non-expanding function has constant density in some link*. We discuss the relation of this bound to prior literature further in Section 5. It remains an interesting open problem whether the expression is tight, or could be improved to match the expansion profile known under stronger expansion guarantees.

As applications of Theorem 1, we give the first structure theorems for simpler combinatorial constructions of high dimensional expanders. We'll start by looking at one of the most classical constructions of simplicial complexes: *clique-complexes*. Given a graph $G = (V, E)$, the k-dimensional clique-complex $K_{G,k}$ is the complex induced by taking the uniform distribution over k-cliques of G. Using Theorem 1, we give the first non-trivial characterization of non-expanding sets on (dense) clique-complexes.

▶ **Corollary 2** (Expansion in Clique-Complexes). *Fix $k \in \mathbb{N}$ and let $G = (V, E)$ be any graph with minimum degree at least $\Delta_{min} \geq \frac{2k-2}{2k-1}|V|$. Then the expansion of any (ε, i)-global boolean function f on the k-dimensional clique-complex $K_{G,k}$ is at least:*

$$\Phi(f) \geq (1 - \mathbb{E}[f])\frac{(i+1)}{k(k-i)} - O(\varepsilon)$$

While Corollary 2 is non-trivial in any dimension, it is strongest in low dimensions. For instance, an interesting implication for 3-clique (triangle) complexes is that any function of triangles with expansion worse than $1/3$ must have a dense vertex.

▶ **Corollary 3** (Low Influence Functions on Triangle Complexes). *Let $G = (V, E)$ be any graph of minimum degree at least $\Delta_{min} \geq \frac{5}{6}|V|$ and $S \subset K_{G,3}(3)$ any set with expansion*

$$\Phi(S) \leq \frac{1}{3} - \delta.$$

Then there exists a vertex $v \in G$ such that S contains at least a $\frac{\delta}{2}$-fraction of the triangles touching v.

[2] Note here we really mean the set of k-faces including some fixed i-face, sometimes called a star (rather than removing the common face as in the formal definition of a link).

Of course, clique-complexes arising from dense graphs are themselves dense objects. A major selling points of prior work was the ability to characterize low-influence functions on several families of *sparse* complexes. While our work cannot match the quantitative strength of these results, it does lead to new insight on the structure of functions on simpler constructions of bounded-degree HDX with quantitatively weaker parameters. In particular, we consider (cutoffs of) the elementary combinatorial HDX of Golowich [40], which tensors a bounded-degree expander G with the complete complex to give a weak bounded-degree two-sided HDX.

▶ **Corollary 4** (Expansion in Combinatorial HDX). *Let X be a $2k$-dimensional "product-HDX" of any graph G and $\Delta_{4k}(2k)$, and $f \in C_k$ any (ε, i)-global boolean function. Then*

$$\Phi(f) \geq \frac{(1-\mathbb{E}[f])i}{(k-1)(k-i)} - O(\varepsilon)$$

As for the case of clique-complexes, this is especially powerful in low dimensions. For concreteness, we again look at the first non-trivial setting of $k = 3$.

▶ **Corollary 5** (Low Influence Functions on Combinatorial HDX). *Let X be a 3-dimensional "product-HDX" of any graph G and $\Delta_n(6)$, and $S \subset X(3)$ be any subset of triangles with expansion at most*

$$\Phi(S) \leq \frac{1}{4} - \delta.$$

Then there exists a vertex $v \in X(1)$ such that S contains a $\Omega(\delta)$ fraction of triangles including the vertex v.

Finally, we note that as an immediate corollary of Theorem 1 we also get a "Kruskal-Katona" type theorem in all the above settings. Kruskal-Katona is a seminal and broadly used result in extremal combinatorics which given a set $S \subset X(k)$, lower bounds the number of $(k-1)$-faces that sit inside S, called its "lower shadow" and denoted ∂S.

▶ **Corollary 6** (Kruskal-Katona for Weak HDX). *Let (X, Π) be a k-dimensional simplicial complex and $S \subset X(k)$ any (ε, i)-global set. Then*

$$\mathbb{E}[\partial S] \geq \mathbb{E}[S]\left(1 + \frac{1-\mathbb{E}[S]}{k-i}\prod_{j=i}^{k-2}(1-\gamma_j) - c_{k,i,\gamma}\varepsilon.\right)$$

where $c_{k,i,\gamma} \leq (k-i+1)\left(\frac{1}{k-i}\prod_{j=i}^{k-2}(1-\gamma_j) - \frac{1}{k-i+1}\prod_{j=i-1}^{k-2}(1-\gamma_j)\right)\left(1+(k-i)\gamma_{i-1}^{(-)}\right)^{-1}$

3.2 Small-Set Expansion

Similar to the setting of the lower walk, links also provide a canonical family of non-expanding sets for the noise operator on simplicial complexes. In this setting, it is typical [56, 44, 7] to use a slightly stronger notion of global functions we'll call "(ε, i)-strongly global", which simply assumes f is sparse on every link:

$$\forall \tau \in X(i) : \mathop{\mathbb{E}}_{X_\tau}[f_\tau] \leq \varepsilon.$$

BHKL [7], GLL [44], and Hopkins [47] prove small-set expansion for strongly global functions on two-sided and one-sided partite HDX with very strong quantitative expansion.[3] While these properties are satisfied by some constructions (e.g. [55]), some variants of the seminal *Ramanujan complexes* (quotients of the affine building) of Lubotzky, Samuels, and Vishne [65] actually fail these conditions.

On the other hand, these complexes exhibit nice *local* structure. In particular, for large enough fields, their links are quantitatively strong partite HDX, and therefore satisfy a small-set expansion theorem. This raises a natural question: is any complex X whose links satisfy global small-set expansion itself a global small-set expander? We answer this question in the affirmative.

▶ **Theorem 7** (Local-to-Global SSE). *Let X be a complex such that $\forall v \in X(1)$:*

1. *Any $(\varepsilon, i-1)$-strongly global function f on X_v has expansion*

$$\Phi_{T_\rho}(f) \geq \phi(\varepsilon, k, i)$$

2. *X_v is a γ-two-sided or γ-one-sided partite HDX*

Then any (ε, i)-global boolean function f has expansion at least

$$\Phi_{T_\rho}(f) \geq \phi(\varepsilon, k, i) - 2^{O(k)}\gamma - 2^{-\Omega(k)}.$$

We note the requirement that links of X are HDX can be removed up to worsened dependence in the dimensional error term (see the full version). The inverse exponential error above is essentially negligible, since any function with respect to the noise operator has non-expansion at least inverse exponential in k.

As an immediate corollary, we prove the Ramanujan complexes are global small-set expanders.

▶ **Corollary 8** (Ramanujan Complexes are Global SSEs). *Let $f \in C_k$ be an (ε, i)-strongly global boolean function on a Ramanujan complex of [65] with $2^{-\Omega(k)}$ one-sided local-spectral expansion. The expansion of f is at least*

$$\Phi_{T_\rho}(f) \geq 1 - \rho^i - O_i(\varepsilon) - 2^{-\Omega(k)}.$$

4 Techniques

We sketch the proofs of Theorem 1 and Theorem 7, starting with the former. The formal versions and proofs of all results are deferred to the full version of the paper.

4.1 Proof Overview of Theorem 1

Our general approach to prove global sets expand is broken into three core components. The first is a new function-dependent variant of "Garland's Lemma", a classical method of breaking global functions into local components over the complex. In particular, we show the expansion of a function f with respect to the lower walks can be written as an expectation of its expansion over links of X.

[3] In particular, they require local-spectral expansion to be inverse exponential in the dimension.

▶ **Lemma 9** (Garland's Lemma for Expansion). *Let (X, Π) be a weighted, pure simplicial complex and $f \in C_k$ any function on k-faces. Then for all $i \leq k$ and $j \leq k - i$:*

$$\Phi(f) = \mathop{\mathbb{E}}_{\tau \sim \pi_j^f}[\Phi_{X_\tau}(f|_\tau)], \tag{1}$$

where $f|_\tau(\sigma) = f(\tau \cup \sigma)$ is the localization of f to the link of τ, and π_j^f is some f-dependent distribution over j-faces of X.

We remark that to the authors' knowledge, this is the first variant of Garland's lemma where the expectation is over a *function-dependent* distribution, rather than simply over the complex distribution π_j, which is necessary for our setting.

The second core component is a variant of the GK's decomposition theorem [42] for the lower walk.

▶ **Theorem 10** (GK-Decomposition). *Let (X, Π) be a k-dimensional simplicial complex. Then for any $f \in C_k$ there is an orthogonal decomposition $f = \sum_{i=0}^{k} f_i$ such that:*

$$\langle N_1^k f, f\rangle \leq \mathbb{E}[f]^2 + \sum_{i=1}^{k-1}\left(1 - \frac{1}{k-i+1}\prod_{j=i-1}^{k-2}(1-\gamma_j)\right)\langle f, f_i\rangle.$$

Finally, the third main component is a "level-1 inequality" showing any $(\varepsilon, 1)$-global function has low projection onto the first level of the GK-decomposition.

▶ **Proposition 11** (Level-1 Inequality). *Let (X, Π) be a k-dimensional complex and f any $(\varepsilon, 1)$-global boolean function $f \in C_k$. Then*

$$\langle f, f_1\rangle \leq \frac{k}{1 + \gamma_0^{(-)}(k-1)}\varepsilon.$$

We remark that this is a corollary of a more general level i inequality that scales with the minimum non-zero eigenvalue of the upper walks (see full version). One could instead apply this directly with the GK decomposition, but the resulting dependence on ε is worse than can be achieved by combining these with Garland's method.

Given these three components, Theorem 1 follows from a careful induction. We first prove the statement directly for $(\varepsilon, 1)$-global functions using the GK-Decomposition and Level-1 inequality. For general i, we use Garland's lemma to reduce to $i = 1$ inside links of X, writing the expansion as $\Phi(f) = \mathbb{E}_{\tau \sim \pi_{i-1}^f}[\Phi_\tau(f|_\tau)]$, and observe that the localization $f|_\tau$ of a (ε, i) global function is itself $(\varepsilon - \delta_\tau, 1)$ global for some $|\delta_\tau| \leq \varepsilon$. Applying the statement for 1-global functions inside the expectation (carefully handling dependence on $\mathbb{E}_{\tau \sim \pi_{i-1}^f}[\delta_\tau]$) then gives the claimed bound.

4.2 Proof Overview of Theorem 7

The first main component of our low influence characterization, Lemma 9, already shows that expansion of the lower walk exhibits local-to-global behavior. The proof of Theorem 7 would follow immediately if the same form of localization were to hold for the noise operator. Unfortunately, while the lemma may be extended to lower walks of arbitrary length, such a statement does not hold directly for the noise operator.

To see why, recall the noise operator is a convex combination of lower walks $T_\rho = \sum_{i=0}^{k} B_k^\rho(i) N_k^{k-i}$ where $B_k^\rho(i) = \binom{k}{i}(1-\rho)^{k-i}\rho^i$. Since expansion is linear, for any function f we can write

$$\Phi_{T_\rho}(f) = \sum_{i=0}^{k} B_k^\rho(i) \Phi_{N_k^{k-i}}(f).$$

Localizing T_ρ naively hits two main issues. First, a walk N_k^i can only be appropriately localized to j-links when $j \le k-i$ (otherwise the walk goes below the level of the link, and cannot be captured at this level of locality). Second, the *coefficients* of T_ρ, which are binomially distributed, depend on dimension and therefore do not localize correctly. In particular, after localizing "permissible" lower walks in the sum, we'd get:

$$\Phi_{T_\rho}(f) = \sum_{i=0}^{j-1} B_k^\rho(i) \Phi_{N_k^i}(f) + \mathbb{E}_{\tau \sim \pi_j^f}\left[\sum_{i=j}^{k} B_k^\rho(i) \Phi_{N_{k-j}^{k-i}}(f|_\tau).\right]$$

To correspond to the local noise operator on the right-hand side, we'd instead need the corresponding coefficients to be $B_{k-j}^\rho(i)$, the binomial distribution on $k-j$ trials, not k trials as in the original instance. A first approach to solving these issues is to define a "shifted" noise operator over the original space that simply throws out the lefthand sum, which is negligible, and replaces the remaining coefficients with $B_{k-j}^\rho(i)$ over the original space. One can then argue that the expansion with respect to the shifted operator "localizes correctly" (allowing us to apply local SSE), and is moreover close to the expansion of the original noise operator. Unfortunately, this procedure results in error scaling with the TV-distance between Binomial distributions on k and $k-j$ trials. This costs $\mathrm{poly}(k^{-1})$ additive error, and therefore does not give the desired global SSE theorem.

To fix this issue, we instead make the stronger assumption that links of X are good two-sided or partite one-sided high dimensional expanders, a property satisfied by our main motivating example (the Ramanujan complexes). This allows us to appeal to the theory of Fourier analysis developed in [19, 6, 7, 44], which shows one may decompose expansion with respect to the lower walks as

$$\Phi_{N_k^i}(f) \approx \frac{1}{\langle f, f\rangle} \sum_{\ell=0}^{k} \lambda(N_k^i)\langle f_\ell, f_\ell\rangle$$

for some decomposition $f = \sum_{i=0}^{k} f_i$ and approximate eigenvalues $\lambda(N_k^i)$. With this in mind, instead of shifting the noise operator, we can decompose the localized expansion in the Fourier basis as

$$\Phi_{T_\rho}(f) \lesssim \sum_{i=0}^{j-1} B_k^\rho(i) + \mathop{\mathbb{E}}_{\tau \sim \pi_j^f}\left[\frac{1}{\langle f|_\tau, f|_\tau\rangle} \sum_{i=j}^{k} B_k^\rho(i) \sum_{\ell=0}^{k-j} \lambda_\ell(N_{k-j}^{k-i}|_\tau)\langle (f|_\tau)_\ell, (f|_\tau)_\ell\rangle\right]$$

Finally for each fixed ℓ, we show its corresponding coefficient in the above sum is at most ρ^ℓ, the (approximate) eigenvalue of the local noise operator itself [7, 44]. This allows us to write the desired localized inequality

$$\Phi_{T_\rho}(f) \lesssim \sum_{i=0}^{j-1} B_k^\rho(i) + \mathop{\mathbb{E}}_{\tau \sim \pi_j^f}\left[\Phi_{T_\rho}(f|_\tau)\right]$$

which may finally be bounded by the local SSE theorem using the fact that any j-restriction $f|_\tau$ of an (ε, i)-strongly global function is $(2\varepsilon, i-j)$-strongly global. The error in this analysis scales only with the cost of cutting off small i (the first term) and the approximation error of the Fourier decomposition. For constant j, the former is at most $\exp(-k)$, while the latter is $2^{O(k)}\gamma$ when j-links are γ-HDX. Thus for complexes with sufficiently expanding links, we get an SSE theorem with negligible additive error as desired.

5 Discussion and Related Work

We close the extended abstract with a more in-depth comparison with prior work and a natural open problem such comparisons raise. At a high level, our work fits into a long line of research examining the structure of low-influence functions beyond the hypercube, starting not long after the KKL Theorem itself with the study of product spaces and the p-biased cube [13, 70, 38, 36, 46, 56] and more recently in extended settings such as the slice [60], multi-slice [35, 69], symmetric group [34], Grassmannian [62, 31], high dimensional expanders [6, 7, 44, 39, 42], and Lie groups [30]. We rely heavily on the Fourier analytic machinery initiated by Dikstein, Dinur, Filmus, and Harsha [19], and Kaufman and Oppenheim [55], and further extended by Bafna, Hopkins, Kaufman, and Lovett [6, 7], Gur Lifshitz, and Liu [44], Hopkins [47], and Gotlib and Kaufman [42].

Concretely, several bounds are known for the expansion of global sets with respect to the lower walk. In the regime of strong (inverse exponential) quantitative expansion, BHKL [6] prove any (ε, i)-global function has expansion at least

$$\Phi(f) \gtrsim (1-\mathbb{E}[f])\frac{i+1}{k} - \binom{k}{i}\varepsilon.$$

This is tight when $\varepsilon \ll \frac{1}{\binom{k}{i}}$, and mimics the structure one would expect to see on general k-fold product spaces. For strongly global functions, BHKL [7], GLL [44], and Hopkins [47] improve the ε-dependence to be *independent of dimension*:

$$\Phi(f) \gtrsim (1-\mathbb{E}[f])\frac{i+1}{k} - 2^{O(i)}\varepsilon.$$

In the regime of weak high dimensional expansion, the only known bound for global functions is due to Gotlib and Kaufman [41], who show that on any *one-sided* HDX:

$$\Phi(f) \geq (1-\mathbb{E}[f])\frac{1}{k-i}\prod_{j=i}^{k-2}(1-\gamma_j) + |X(k)|\varepsilon.$$

Unlike the prior bounds, however, this is only really meaningful when $\varepsilon \approx 0$, a non-trivial but extremely strong notion of globalness that does not lead to meaningful structural characterizations.

In some sense our bound, Theorem 1, sits between the latter two. it has the beneficial properties of the latter in that it holds for global functions and under arbitrary (two-sided) local-spectral expansion, and of the former in that the dependence on ε is typically constant. On the other hand, due to being obtained through an inductive approach, it also inherits the main term of Gotlib-Kaufman [42], scaling with $\frac{1}{k-i}$ instead of $\frac{i+1}{k}$. When i and k are large, the former dependence is substantially worse.

At a technical level, this difference in main term stems from the fact that our "advantage" for i-global functions, like in Gotlib-Kaufman [42], comes from the second eigenvalue of the lower walk inside i-links of X, which scales as $1-\frac{1}{k-i+1}\prod_{j=i-1}^{k-2}(1-\gamma_j)$. On the other hand,

on very strong HDX, the advantage of i-global functions comes from the ith approximate eigenvalue of the global lower walk on X itself, which scales with $1 - \frac{i}{k}$. It is natural to conjecture the "correct" dependence sits between these two bounds, as $1 - \frac{i}{k} \prod_{j=i-1}^{k-2} (1 - \gamma_j)$, closer to the type of bound known for the second eigenvalue from [3]. Unfortunately, both the techniques of this paper and Fourier analytic techniques from prior work (which typically incur exponential error) seem far from proving such a bound.

References

1 Rudolf Ahlswede and Peter Gács. Spreading of sets in product spaces and hypercontraction of the markov operator. *The annals of probability*, pages 925–939, 1976.

2 Vedat Levi Alev, Fernando Granha Jeronimo, and Madhur Tulsiani. Approximating constraint satisfaction problems on high-dimensional expanders. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 180–201. IEEE, 2019. doi:10.1109/FOCS.2019.00021.

3 Vedat Levi Alev and Lap Chi Lau. Improved analysis of higher order random walks and applications. In *Proceedings of the 52nd annual ACM SIGACT symposium on theory of computing*, pages 1198–1211, 2020. doi:10.1145/3357713.3384317.

4 Nima Anari, Kuikui Liu, and Shayan Oveis Gharan. Spectral independence in high-dimensional expanders and applications to the hardcore model. *SIAM Journal on Computing*, 53(6):FOCS20–1, 2021.

5 Nima Anari, Kuikui Liu, Shayan Oveis Gharan, and Cynthia Vinzant. Log-concave polynomials ii: high-dimensional walks and an FPRAS for counting bases of a matroid. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 1–12, 2019. doi:10.1145/3313276.3316385.

6 Mitali Bafna, Max Hopkins, Tali Kaufman, and Shachar Lovett. High dimensional expanders: Eigenstripping, pseudorandomness, and unique games. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1069–1128. SIAM, 2022. doi:10.1137/1.9781611977073.47.

7 Mitali Bafna, Max Hopkins, Tali Kaufman, and Shachar Lovett. Hypercontractivity on high dimensional expanders. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 185–194, 2022. doi:10.1145/3519935.3520040.

8 Mitali Bafna, Noam Lifshitz, and Dor Minzer. Constant degree direct product testers with small soundness. In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 862–869. IEEE, 2024. doi:10.1109/FOCS61266.2024.00059.

9 Mitali Bafna, Dor Minzer, Nikhil Vyas, and Zhiwei Yun. Quasi-linear size pcps with small soundness from hdx. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 45–53, 2025. doi:10.1145/3717823.3718197.

10 Boaz Barak, Pravesh K. Kothari, and David Steurer. Small-Set Expansion in Shortcode Graph and the 2-to-2 Conjecture. In *10th Innovations in Theoretical Computer Science Conference (ITCS 2019)*, volume 124 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 9:1–9:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ITCS.2019.9.

11 Daniel Beaglehole, Max Hopkins, Daniel Kane, Sihan Liu, and Shachar Lovett. Sampling equilibria: Fast no-regret learning in structured games. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3817–3855. SIAM, 2023. doi:10.1137/1.9781611977554.CH149.

12 Antonio Blanca, Pietro Caputo, Zongchen Chen, Daniel Parisi, Daniel Štefankovič, and Eric Vigoda. On mixing of markov chains: Coupling, spectral independence, and entropy factorization. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3670–3692. SIAM, 2022. doi:10.1137/1.9781611977073.145.

13 Jean Bourgain, Jeff Kahn, Gil Kalai, Yitzhak Katznelson, and Nathan Linial. The influence of variables in product spaces. *Israel Journal of Mathematics*, 77(1):55–64, 1992.

14 Zongchen Chen, Andreas Galanis, Daniel Štefankovič, and Eric Vigoda. Rapid mixing for colorings via spectral independence. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1548–1557. SIAM, 2021.

15 Zongchen Chen, Kuikui Liu, and Eric Vigoda. Optimal mixing of glauber dynamics: Entropy factorization via high-dimensional expansion. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1537–1550, 2021. doi:10.1145/3406325.3451035.

16 Zongchen Chen, Kuikui Liu, and Eric Vigoda. Rapid mixing of glauber dynamics up to uniqueness via contraction. *SIAM Journal on Computing*, 52(1):196–237, 2023. doi:10.1137/20M136685X.

17 Yotam Dikstein. New high dimensional expanders from covers. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 826–838, 2023. doi:10.1145/3564246.3585183.

18 Yotam Dikstein and Irit Dinur. Agreement testing theorems on layered set systems. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1495–1524. IEEE, 2019. doi:10.1109/FOCS.2019.00088.

19 Yotam Dikstein, Irit Dinur, Yuval Filmus, and Prahladh Harsha. Boolean Function Analysis on High-Dimensional Expanders. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018)*, volume 116 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.38.

20 Yotam Dikstein, Irit Dinur, and Alexander Lubotzky. Low acceptance agreement tests via bounded-degree symplectic hdxs. In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 826–861. IEEE, 2024. doi:10.1109/FOCS61266.2024.00058.

21 Yotam Dikstein and Max Hopkins. Chernoff bounds and reverse hypercontractivity on hdx. In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 870–919. IEEE, 2024. doi:10.1109/FOCS61266.2024.00060.

22 Yotam Dikstein, Max Hopkins, Russell Impagliazzo, and Toniann Pitassi. High rate efficient local list decoding from hdx. *arXiv preprint*, 2026. doi:10.48550/arXiv.2601.22535.

23 Irit Dinur, Shai Evra, Ron Livne, Alexander Lubotzky, and Shahar Mozes. Locally testable codes with constant rate, distance, and locality. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 357–374, 2022. doi:10.1145/3519935.3520024.

24 Irit Dinur, Yuval Filmus, Prahladh Harsha, and Madhur Tulsiani. Explicit sos lower bounds from high-dimensional expanders. In *12th Innovations in Theoretical Computer Science Conference, ITCS 2021*, pages 1–16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, 2021. doi:10.4230/LIPIcs.ITCS.2021.38.

25 Irit Dinur and Tali Kaufman. High dimensional expanders imply agreement expanders. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 974–985. IEEE, 2017. doi:10.1109/FOCS.2017.94.

26 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. On non-optimally expanding sets in grassmann graphs. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 940–951, 2018. doi:10.1145/3188745.3188806.

27 Irit Dinur, Subhash Khot, Guy Kindler, Dor Minzer, and Muli Safra. Towards a proof of the 2-to-1 games conjecture? In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 376–389, 2018. doi:10.1145/3188745.3188804.

28 Irit Dinur, Siqi Liu, and Rachel Yun Zhang. New Codes on High Dimensional Expanders. In *40th Computational Complexity Conference (CCC 2025)*, volume 339 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 27:1–27:42. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.CCC.2025.27.

29 David Ellis, Guy Kindler, and Noam Lifshitz. An analogue of bonami's lemma for functions on spaces of linear maps, and 2-2 games. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 656–660, 2023. doi:10.1145/3564246.3585116.

30 David Ellis, Guy Kindler, Noam Lifshitz, and Dor Minzer. Product mixing in compact lie groups. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 1415–1422, 2024. doi:10.1145/3618260.3649626.

31 David C Ellis, Guy Kindler, and Noam Lifshitz. Forbidden intersection problems for families of linear maps. *Discrete Analysis*, 19, 2023.

32 Yumou Fei, Dor Minzer, and Shuo Wang. Multi-pass streaming lower bounds for approximating max-cut. In *2025 IEEE 66th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1537–1560, 2025. doi:10.1109/FOCS63196.2025.00081.

33 Weiming Feng, Heng Guo, Yitong Yin, and Chihao Zhang. Rapid mixing from spectral independence beyond the boolean domain. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1558–1577. SIAM, 2021. doi:10.1137/1.9781611976465.95.

34 Yuval Filmus, Guy Kindler, Noam Lifshitz, and Dor Minzer. Hypercontractivity on the symmetric group. In *Forum of Mathematics, Sigma*, volume 12, page e6. Cambridge University Press, 2024.

35 Yuval Filmus, Ryan O'Donnell, and Xinyu Wu. Log-sobolev inequality for the multislice, with applications. *Electronic Journal of Probability*, 27:1–30, 2022.

36 Ehud Friedgut. Boolean functions with low average sensitivity depend on few coordinates. *Combinatorica*, 18(1):27–35, 1998. doi:10.1007/PL00009809.

37 Ehud Friedgut and Jean Bourgain. Sharp thresholds of graph properties, and the k-sat problem. *Journal of the American mathematical Society*, 12(4):1017–1054, 1999.

38 Ehud Friedgut and Gil Kalai. Every monotone graph property has a sharp threshold. *Proceedings of the American mathematical Society*, 124(10):2993–3002, 1996.

39 Jason Gaitonde, Max Hopkins, Tali Kaufman, Shachar Lovett, and Ruizhe Zhang. Eigenstripping, Spectral Decay, and Edge-Expansion on Posets. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2022)*, volume 245 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 16:1–16:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.APPROX/RANDOM.2022.16.

40 Louis Golowich. Improved Product-Based High-Dimensional Expanders. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2021)*, volume 207 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.APPROX/RANDOM.2021.38.

41 Roy Gotlib and Tali Kaufman. Beyond the worst case: Structured convergence of high dimensional random walks. *CoRR*, 2022. doi:10.48550/arXiv.2208.03241.

42 Roy Gotlib and Tali Kaufman. Fine grained analysis of high dimensional random walks. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, 2023.

43 Ben Green and Mehtaab Sawhney. New bounds for the Furstenberg-Sárközy theorem. *arXiv preprint*, 2024. arXiv:2411.17448.

44 Tom Gur, Noam Lifshitz, and Siqi Liu. Hypercontractivity on high dimensional expanders. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 176–184, 2022. doi:10.1145/3519935.3520004.

45 Tom Gur, Dor Minzer, Guy Weissenberg, and Kai Zhe Zheng. 3-query rldcs are strictly stronger than 3-query ldcs. *arXiv preprint*, 2025. doi:10.48550/arXiv.2512.12960.

46 Hamed Hatami. A structure theorem for boolean functions with small total influences. *Annals of Mathematics*, pages 509–533, 2012.

47 Max Hopkins. Hypercontractivity on hdx ii: Symmetrization and q-norms. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, pages 96–103, 2025. doi:10.1145/3717823.3718187.

48 Max Hopkins and Ting-Chun Lin. Explicit lower bounds against ω (n)-rounds of sum-of-squares. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 662–673. IEEE, 2022. doi:10.1109/FOCS54457.2022.00069.

49 Jun-Ting Hsieh, Alexander Lubotzky, Sidhanth Mohanty, Assaf Reiner, and Rachel Yun Zhang. Explicit lossless vertex expanders. In *2025 IEEE 66th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 894–911, 2025. doi:10.1109/FOCS63196.2025.00046.

50 Vishesh Jain, Huy Tuan Pham, and Thuy-Duong Vuong. Spectral independence, coupling, and the spectral gap of the glauber dynamics. *Information Processing Letters*, 177:106268, 2022. doi:10.1016/j.ipl.2022.106268.

51 J. Kahn, G. Kalai, and N. Linial. The influence of variables on boolean functions. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 68–80, Los Alamitos, CA, USA, October 1988. IEEE Computer Society. doi:10.1109/SFCS.1988.21923.

52 Tali Kaufman, David Kazhdan, and Alexander Lubotzky. Isoperimetric inequalities for ramanujan complexes and topological expanders. *Geometric and Functional Analysis*, 26(1):250–287, 2016.

53 Tali Kaufman and David Mass. Local-To-Global Agreement Expansion via the Variance Method. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 74:1–74:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ITCS.2020.74.

54 Tali Kaufman and Izhar Oppenheim. Construction of new local spectral high dimensional expanders. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 773–786, 2018. doi:10.1145/3188745.3188782.

55 Tali Kaufman and Izhar Oppenheim. High order random walks: Beyond spectral gap. *Combinatorica*, pages 1–37, 2020.

56 Peter Keevash, Noam Lifshitz, Eoin Long, and Dor Minzer. Hypercontractivity for global functions and sharp thresholds. *Journal of the American Mathematical Society*, 37(1):245–279, 2024.

57 Peter Keevash, Noam Lifshitz, and Dor Minzer. On the largest product-free subsets of the alternating groups. *Inventiones mathematicae*, 237(3):1329–1375, 2024.

58 Nathan Keller, Noam Lifshitz, and Omri Marcus. Sharp hypercontractivity for global functions. *Journal of the European Mathematical Society*, 2026. Published online first. doi:10.4171/JEMS/1762.

59 Nathan Keller, Noam Lifshitz, Dor Minzer, and Ohad Sheinfeld. On t-intersecting families of permutations. *Advances in Mathematics*, 445:109650, 2024.

60 Subhash Khot, Dor Minzer, Dana Moshkovitz, and Muli Safra. Small-set expansion in the johnson graph. *Theory of Computing*, 21(1):1–43, 2025. doi:10.4086/TOC.2025.V021A002.

61 Subhash Khot, Dor Minzer, and Muli Safra. On independent sets, 2-to-2 games, and grassmann graphs. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 576–589, 2017. doi:10.1145/3055399.3055432.

62 Subhash Khot, Dor Minzer, and Muli Safra. Pseudorandom sets in grassmann graph have near-perfect expansion. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 592–601. IEEE, 2018. doi:10.1109/FOCS.2018.00062.

63 Noam Lifshitz and Dor Minzer. Noise sensitivity on the p-biased hypercube. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1205–1226. IEEE, 2019. doi:10.1109/FOCS.2019.00075.

64 Kuikui Liu. From Coupling to Spectral Independence and Blackbox Comparison with the Down-Up Walk. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2021)*, volume 207 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 32:1–32:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.APPROX/RANDOM.2021.32.

65 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Explicit constructions of ramanujan complexes of type Ad. *European Journal of Combinatorics*, 26(6):965–993, 2005. doi:10.1016/J.EJC.2004.06.007.

66 Dor Minzer and Kai Zhe Zheng. Near optimal alphabet-soundness tradeoff pcps. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 15–23, 2024. doi:10.1145/3618260.3649606.

67 Izhar Oppenheim. Local spectral expansion approach to high dimensional expanders part i: Descent of spectral gaps. *Discrete & Computational Geometry*, 59(2):293–330, 2018. doi:10.1007/S00454-017-9948-X.

68 Pavel Panteleev and Gleb Kalachev. Asymptotically good quantum and locally testable classical ldpc codes. In *Proceedings of the 54th annual ACM SIGACT symposium on theory of computing*, pages 375–388, 2022. doi:10.1145/3519935.3520017.

69 Justin Salez. A sharp log-sobolev inequality for the multislice. *Annales Henri Lebesgue*, 4:1143–1161, 2021.

70 Michel Talagrand. On Russo's approximate zero-one law. *The Annals of Probability*, pages 1576–1587, 1994.

Improved Local Computation of Edge Orientation

Reut Levi
Reichman University, Herzliya, Israel

Bar Rushkin
Reichman University, Herzliya, Israel

Abstract

In this paper, we study the problem of orienting the edges of a graph G so that every vertex has bounded out-degree in the *local computation algorithms (LCA)* model, as defined by Rubinfeld et al. (ICS 2011). More specifically, given a query $e \in E$ our algorithm returns the orientation of e such that with high constant probability (namely, at least 0.9) the out-degree of each vertex is bounded by r where r is a parameter. We provide such an upper bound for any $r = \Omega(arb(G) \cdot \log n)$ where $arb(G)$ denotes the arboricity of G (we note that such orientation exists only when $r = \Omega(arb(G))$). Our query complexity is $\tilde{O}(n \cdot arb(G)/r^2)$ in the worst case and $O(1)$ on expectation (over the vertices and the randomness of the algorithm). This generalizes the upper bound by Mitrović–Rubinfeld–Singhal (ESA 2024) that provided a similar upper bound only when $r = \Omega((arb(G)^2 \cdot n))^{1/3}$.

For $r = \Omega(arb(G) \cdot \log n)$, our algorithm also improves their weaker upper bound of $\tilde{O}(n/r)$ queries for the special case where the input graph is a tree (whose arboricity is 1).

Our algorithm assigns each vertex a *level* derived from locally sampled neighborhoods combined through a staggered multi-scale recursion, inspired by the recent arboricity-approximation framework of Dai–Ghaffari–Portmann (FOCS 2025). The main novelty of our approach is that it reconstructs levels consistently across the graph while simultaneously respecting the out-degree bound and maintaining locality of computation.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms; Mathematics of computing → Graph algorithms; Theory of computation → Graph algorithms analysis

Keywords and phrases Local Algorithms, Sublinear-time Algorithms, Edge Orientation, Bounded Arboricity

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.75

Category RANDOM

Funding *Reut Levi*: The author was supported by the Israel Science Foundation under Grant 1867/20.
Bar Rushkin: The author was supported by the Israel Science Foundation under Grant 1867/20.

1 Introduction

Orienting graph edges while satisfying degree constraints has broad algorithmic applications. From a structural perspective, bounded out-degree orientations capture the sparsity of a graph via its degeneracy and arboricity, and play a key role in graph coloring and subgraph enumeration algorithms [4, 13].

Due to this fact, low-out-degree orientation has been extensively studied in the dynamic, distributed, and massively parallel settings (see e.g. [10, 17, 9, 7, 3]).

In many local settings, a standard approach is to simulate a distributed peeling-type process that, for a graph G of arboricity $\alpha = arb(G)$, yields an orientation with maximum out-degree at most $\alpha(2 + \epsilon)$ in $O(\log n)$ iterations.


© Reut Levi and Bar Rushkin;
licensed under Creative Commons License CC-BY 4.0



Mitrović, Rubinfeld, and Singhal [14] initiated the study of computing bounded out-degree edge orientations in the LCA model. They showed that directly simulating the peeling approach via standard LCA transformations can require $\Omega(n)$ probes per query. They also proved a lower bound of $\Omega(\sqrt{n}/r)$ probes per query for producing an r-out-degree orientation, even when the input graph is a forest (i.e., when $\alpha = 1$).

On the algorithmic side, they gave sublinear-probe LCAs for forests and for general graphs. For general graphs, however, their approach requires $r = \Omega\big((\alpha^2 n)^{1/3}\big)$, and for this range they provide a randomized LCA that computes an r-orientation using $\tilde{O}(\max\{\alpha n/r^2, 1\})$ probes per query. Their approach is based on a degree-threshold decomposition together with randomized sampling of the immediate neighborhood of a vertex, rather than on recursively reconstructing a global peeling process.

As a result, their work leaves a gap both in probe complexity and in the range of parameters for which sublinear algorithms are known. In this paper, we address the latter gap, up to logarithmic factors, by providing an LCA with the same probe complexity for all $r = \Omega(\alpha \log n)$. Since r must be at least $\Omega(\alpha)$, this extends the best known upper bound essentially down to the smallest range currently compatible with the sampling requirements of the approach.

1.1 The problem of r-orientation and the peeling process

A natural *global* approach for achieving almost optimal out-degree is the *peeling process*. This process proceeds in iterations. In each iteration, i, all vertices whose *current* degree is at most $(2+\epsilon)\alpha$ are removed from the graphs. These vertices will be called the i-th layer. The process repeats until no vertices remain. Edges are then oriented from earlier to later layers, and arbitrarily within each layer, resulting in a maximum out-degree of $(2+\epsilon)\alpha$ after $O(\log n)$ phases. However, when simulated in the LCA model using the standard Parnas–Ron simulation argument, peeling requires $\Omega(n)$ probes per query, and is therefore not sublinear [15].

In a pair of papers studying the problem of approximating the arboricity of a graph, the authors nevertheless manage to locally simulate a peeling process. In particular, both the algorithm of Eden, Mossel, and Ron [6] and that of Dai, Ghaffari, and Portmann [5] employ a local *simulation of a peeling process*. However, in both cases the simulation does not reconstruct the order in which vertices are peeled. Moreover, the local views produced by the simulation do not correspond to any consistent global peeling sequence.

EMR [6]: pruning-based simulation

In this paper, Eden, Mossel, and Ron [6] introduced a local recursive procedure designed to estimate whether a global peeling process removes a vertex v by the i-th iteration. To keep query complexity manageable, the algorithm prunes branches of the recursion that are expected to be costly. While their analysis shows that this pruning does not significantly distort the simulation's outcome, the procedure has a key limitation for our purposes: it does not recover the actual peeling iteration, but only provides an upper bound. More critically, the local views generated by the simulation do not correspond to any coherent global peeling order. Specifically, on query v, the algorithm attempts to peel a sample consisting of a $\Theta(1/\alpha)$-fraction of the neighbors of v, while *pruning* those predicted to trigger deep recursion. This technique controls the expected recursion depth and yields an $O(\log^2 n)$-approximation using $\tilde{O}(n/\alpha)$ probes. However, the resulting simulation should be viewed only as an approximation of the ideal peeling process. In particular, the local

simulation used for one vertex may correspond to one approximate peeling process, while the simulation for another vertex may correspond to a different approximate peeling process. As a result, the algorithm does not recover the actual peeling iterations and does not define a globally consistent peeling order, which requires a different approach in our setting.

DGP [5]: staggered multi-copy simulation

In their recent work, Dai, Ghaffari, and Portmann [5] propose a procedure that estimates whether a vertex would be peeled in a global peeling process. Their method samples a set of vertices and, for each one, simulates a peeling process to determine if it gets removed – but crucially, it does not recover the iteration in which the vertex is peeled. To manage query complexity, they avoid pruning and instead run multiple simulations in parallel, each with a different pace. Their key insight is that if the graph has low arboricity then with high probability, one of these simulations will terminate quickly and return an answer. Moreover, they design a clever scheduling scheme that ensures the algorithm reaches this efficient simulation without incurring a significant overhead in queries.

However, this approach has a fundamental limitation for our purposes: each simulation uses independent randomness, which is essential for their analysis but prevents the results from aligning with any consistent global peeling order. In particular, the different simulations do not correspond to different executions of the same ideal peeling process, but rather to different approximate local simulations of peeling. Consequently, the depth of the first recursive exploration that terminates successfully should not be interpreted as the actual peeling level of the vertex.

To illustrate this, consider two adjacent vertices u and v, where u is peeled before v in the ideal peeling process. Due to the independent randomness used in the local simulations, it may happen that the recursive exploration rooted at u encounters several bad samples and only terminates after many recursive levels, while the exploration rooted at v quickly encounters a good sample and terminates much earlier. Thus, the local recursion depth may suggest that u belongs to a later layer than v, even though the ideal peeling order is the opposite. Since these local simulations are generated independently, there is no globally consistent peeling order that they collectively realize.

Moreover, the algorithm only determines whether a vertex is peeled – not when. This lack of global coherence motivates the need for a different strategy in our orientation setting.

1.2 Our result

We present a new local computation algorithm for r-orientation in general graphs. Given a query access to an input graph $G = (V, E)$ of arboricity α and a parameter $r = \Omega(\alpha \log n)$, the algorithm answers each edge-orientation query using $O\left(\frac{\alpha n \log^2 n}{r^2}\right)$ probes, and with high constant probability produces an orientation of maximum out-degree r. In particular, we prove the following theorem.

▶ **Theorem 1** (LCA for bounded out-degree orientation). *Let $G = (V, E)$ be an n-vertex graph of arboricity $\alpha = arb(G)$, given through adjacency-list probe access. Let r be a parameter satisfying $r \geq C\alpha \log n$ for a sufficiently large universal constant C. Then there exists a randomized LCA which, on query an edge $\{u, v\} \in E$, returns an orientation of this edge such that, with high constant probability over the random tape, all answers are consistent with a single orientation of G whose maximum out-degree is at most r. The probe complexity per edge query is $O\left(\frac{\alpha n \log^2 n}{r^2}\right)$.*

To compare the parameter ranges, consider the case $\alpha = O(1)$. The previous upper bound of Mitrović, Rubinfeld, and Singhal [14] achieves probe complexity $\tilde{O}(n\alpha/r^2)$, but only for $r = \Omega(n^{1/3})$. Our result achieves the same asymptotic upper bound for every $r = \Omega(\log n)$. In particular, for $r = n^{1/4}$, which is outside the range of the previous general-graph algorithm, our LCA uses $O(n^{1/2} \log^2 n)$ probes per query and produces an orientation of maximum out-degree $n^{1/4}$. Moreover, in this regime the lower bound of Mitrović, Rubinfeld, and Singhal [14] is already $\Omega(n^{1/4})$, even for trees, which are in some sense the simplest constant-arboricity graphs. Thus, the improvement in the range of r is meaningful even in parameter regimes where the probe complexity remains strongly sublinear.

1.3 Description of our algorithm

At a high level, our LCA consists of three components. The first assigns each vertex a level by locally simulating a peeling process. The second answers an edge query $\{u, v\}$ by comparing the levels of u and v, orienting the edge from the lower-level endpoint to the higher-level endpoint (breaking ties according to vertex IDs). The third component implements a single local simulation step of the peeling process.

We begin with the intuition behind the local simulation. Consider the ideal peeling process in which, at every iteration, all vertices of degree at most $\Theta(\alpha)$ are removed. A vertex belongs to layer t if it is removed during the t-th iteration. Observe that if one could recursively follow only neighbors belonging to strictly lower layers, then any recursive exploration rooted at a vertex in layer t would have depth at most t.

However, the algorithm does not know the layers in advance. Instead, for each vertex v and iteration t, the algorithm generates random samples of the neighbors of v, where each neighbor is included independently with probability $\Theta(\log n/r)$. Intuitively, if many neighbors of v belong to lower layers, then with constant probability, a sampled set contains only such neighbors. In this case, recursively exploring the sampled neighbors simulates progress toward lower peeling layers.

A single local simulation step proceeds as follows. Initially, the algorithm queries the degree of v. If the degree is at most $\Theta(\alpha)$, the the first peeling layer. Otherwise, the algorithm recursively explores sampled neighbors while decreasing the iteration counter. The recursion terminates if the iteration counter reaches zero, if a low-degree vertex is encountered, or if the total number of probes exceeds the target budget $\tilde{O}(n\alpha/r^2)$.

One difficulty is that a random sample may contain neighbors that do not belong to lower layers, causing the recursive exploration to continue for much longer than the ideal peeling depth. As a mental experiment, we first assume that this does not happen, and explain why the query complexity can be bounded under this assumption. We then explain how this assumption can be removed using the staggered scheduling technique of Dai, Ghaffari, and Portmann [5]. Finally, we explain why introducing the structure of iterations is necessary in order to obtain a globally coherent solution.

Step 1: Assuming one good sample per vertex

To provide intuition for the query bound, let us first consider an idealized setting in which every vertex v is associated with a single *good* sample, namely a sample containing only neighbors that belong to strictly lower peeling layers than v. For the moment, we ignore the staggered scheduling and assume that the algorithm recursively explores only this single sample.

A natural way to analyze the query complexity would be to bound the size of the recursive exploration tree rooted at every vertex. Instead, we reverse the perspective and ask the following question: for a fixed vertex u, how much does u contribute to the recursive trees rooted at other vertices?

Since samples are good, every recursive call moves to a strictly lower peeling layer. Thus every root-to-node trajectory in the recursive exploration corresponds to a directed path in the graph, where edges are directed from higher layers to lower layers. By the peeling property, every vertex has out-degree at most 3α in this directed graph.

Fix a vertex u. Every time u appears in the recursive tree of another vertex, there must exist a sampled directed path leading to u. The expected number of sampled directed paths of length j ending at u is at most $(3\alpha)^j p^j = (3\alpha p)^j$, because there are at most $(3\alpha)^j$ directed paths of length j leading to u, and each such path survives in the recursive exploration with probability p^j. Therefore, the expected total contribution of u to all recursive trees is at most $1 + \sum_{j\geq 1}(3\alpha p)^j$.

Since $p = \Theta(\log n/r)$ and $r = x\alpha$, we have $3\alpha p = \tilde{O}(1/x)$. For sufficiently large x, this geometric series is bounded by $1 + \tilde{O}(1/x)$.

Summing over all vertices, the expected total additional recursive work is therefore $\tilde{O}(n/x)$. Consequently, by Markov's inequality, only a small number of vertices can have recursive explorations whose size exceeds the probe budget. These vertices are the overflow vertices. Since the overflow threshold is set to $\tilde{O}(n\alpha/r^2)$, the expected total work bound implies that, with constant probability, the number of overflow vertices is at most $O(r)$, and the constants are chosen so that it is at most $r/2$.

Step 2: Why staggered scheduling is needed

The discussion above assumed that every sample is good. In reality, however, a random sample is good only with constant probability. Thus, in order to find a good sample, the algorithm performs $O(\log n)$ independent sampling trials for every vertex.

A naive implementation would recursively explore all these trials, increasing the query complexity by an additional logarithmic factor *at every* recursive level. Instead, following Dai, Ghaffari, and Portmann [5], we evaluate these trials according to a staggered schedule.

The key idea is that trial i is slowed down by a factor 2^i, namely it is advanced only once every 2^i scheduling steps. Since each trial is good with constant probability, the index of the first good trial is geometrically distributed. Consequently, although there are $O(\log n)$ possible trials, the expected amount of work invested before reaching the first good one remains bounded by a convergent geometric sum. Intuitively, the algorithm tries many possible recursive explorations, while paying essentially only for the first successful one.

As a result, the accounting argument from the idealized setting still applies, up to a constant-factor overhead in expectation.

Step 3: Why iterations are needed

The discussion above explains why the recursive explorations remain small in expectation. We now turn to the correctness argument, namely why the resulting levels induce an orientation in which every vertex has at most r neighbors whose level is greater than or equal to its own.

The key point is that the algorithm should mimic the ideal peeling process. Suppose that a vertex v still has many neighbors that were not peeled after t iterations of the ideal process. Then we would like to conclude that v itself should also survive iteration $t + 1$. Intuitively, this follows because every sample of v is likely to contain at least one neighbor that survived iteration t, preventing the recursive exploration from terminating too early.

For this argument to hold, however, the randomness used to determine whether a neighbor survives iteration t must be independent of the randomness used when evaluating v in iteration $t+1$. Otherwise, the events become circularly dependent: the randomness used to decide whether u survives could simultaneously influence whether v observes u, destroying the inductive peeling argument.

For this reason, the algorithm uses different random samples for different iterations. Intuitively, iteration t determines which vertices survive after t rounds of peeling, while iteration $t+1$ uses fresh randomness to test whether a vertex still has many neighbors that survived iteration t. This separation between iterations is what allows us to argue inductively that vertices with many surviving neighbors cannot themselves be peeled too early.

This should be contrasted with the approach of Dai, Ghaffari, and Portmann [5], where independent recursive simulations are used only to decide whether a vertex is peeled. Since those simulations are not meant to define a common level assignment, they may use fresh randomness across recursive calls. In our setting, however, the randomness associated with each vertex and each iteration is fixed globally and reused across all recursive calls and all edge queries. This is crucial for ensuring that the local computations are consistent with one global level assignment.

1.4 Why the DGP batching technique does not remove the $\log n$ Gap

Dai, Ghaffari, and Portmann [5] strengthened the staggered scheduling framework by introducing *batches* of parallel recursive simulations. Each batch contains many independent recursive calls, and a vertex is considered peeled once a majority of the recursive calls in some batch terminate. The role of these batches is to amplify correctness in the high-arboricity case: if a vertex belongs to a dense core, then with high probability most recursive calls do not terminate, preventing the vertex from being mistakenly declared peelable. This amplification allows them to obtain a constant-factor approximation.

In our setting, however, this batching mechanism does not seem useful. First, the arboricity bound is already given, and thus we do not need to distinguish between low- and high-arboricity graphs. More importantly, the logarithmic gap between r and α does not stem from the probability amplification aspect of the recursion, but rather from the need to detect surviving neighbors with high probability in every iteration.

Specifically, to ensure the out-degree bound, if a vertex has many neighbors that survived previous iterations, then its samples in the next iteration must hit one of these surviving neighbors with high probability. This requires $p = \Omega(\log n/r)$. At the same time, the query complexity analysis requires the branching factor to remain bounded, namely that αp is bounded away from one. Together, these conditions imply $r = \Omega(\alpha \log n)$. Since the batching mechanism amplifies correctness but does not reduce the sampling probability p, it does not seem to improve the parameter range in our orientation application.

1.5 Related Work

Local Computation Algorithms (LCAs) answer *local* queries about a global solution using a sublinear number of input probes per query, and the answers over all queries are consistent with one global solution [16, 1].

By now, a wide range of results have been established within the LCA model, including maximal independent set [16], graph coloring [8], approximate maximum matching [12], and minor-free graph partitioning [11]. Edge orientations [14] are among the more recent additions to this growing body of work. In particular Mitrović, Rubinfeld, and Singhal [14] obtained the following results.

- *Lower bound.* Any (possibly randomized) LCA that outputs an r-orientation with success probability at least 0.9 must make $\Omega(n^{1/2}/r)$ probes in the worst case, even on forests ($\alpha = 1$) and even when allowed adjacency-list, adjacency-matrix, or degree queries.
- *Upper bounds for general graphs.* When $r \geq 10(\alpha^2 n)^{1/3}$, they obtain probe complexity $\tilde{O}(\max\{\alpha n/r^2, 1\})$.
- *General forests.* When the graph is a forest they obtain an upper bound of $\tilde{O}(n/r)$ queries for any r.
- *Bounded-degree forests.* For forests of maximum degree Δ they provide an LCA with probe complexity $\Delta\, n^{\,1-\log_\Delta r+o(1)}$.

2 Preliminaries

Let $G = (V, E)$ be a simple graph on $n = |V|$ vertices. For $S \subseteq V$ we denote by $E(S)$ the set of edges with both endpoints in S, and by $G[S]$ the induced subgraph on S. The degree of a vertex v is denoted $d(v)$.

▶ **Definition 2** (Arboricity). *The* arboricity *of G, denoted α, is the smallest integer k for which E can be partitioned into k forests. Equivalently,*

$$\alpha = \max_{S \subseteq V,\, |S| \geq 2} \left\lceil \frac{|E(S)|}{|S| - 1} \right\rceil .$$

Graph access model. We assume adjacency-list probe access to the input graph $G = (V, E)$ where $V = [n]$. Specifically, the probe $\texttt{neighbor}(v, i)$ returns the i-th neighbor or $\perp$ if it has less than i neighbors. We also use degree probes, where $\texttt{degree}(v)$ returns the degree of v.

The complexity of the LCA is mainly measured by the number of probes (query complexity), running time per query, and the amount of randomness used.

▶ **Definition 3** (Local Computation Algorithm (LCA) - [16]). *Let $\mathcal{P}$ be a computational problem where, given an input x of size n, the goal is to compute a valid solution $y \in F(x)$. A randomized algorithm $\mathcal{A}$ is a* ***(t, s, δ)-local computation algorithm*** *for $\mathcal{P}$ if, when given probe access to input x and random tape r (the random tape r is fixed once and reused across all queries):*

1. ***Query Handling:*** *For any query q_i (representing a specific part of the solution), $\mathcal{A}(q_i, x, r)$ returns a partial answer y_{q_i}.*
2. ***Sublinear Complexity:*** *The probe complexity per query is sublinear in n.*
3. ***Consistency:*** *The algorithm is* consistent *with a single global valid solution $y \in F(x)$. That is, for any sequence of queries $q_1, q_2, \ldots, q_m$, the algorithm produces an output sequence $y_{q_1}, y_{q_2}, \ldots, y_{q_m}$ that matches the values in y at those locations.*
4. ***Correctness:*** *For all x, with probability at least $2/3$ (over the random tape r) [1], the algorithm answers all queries consistently with a valid solution.*

2.1 r-Orientation in the LCA model

An LCA for the problem of r-orientation receives queries of the form "what is the orientation of edge $\{u, v\}$?" and must answer with either $u \to v$ or $v \to u$, using a number of probes that is sublinear in n, while ensuring that all such answers are consistent with a single global orientation whose maximum out-degree is at most r.

[1] Alternatively, one may require high success probability, namely, $1 - 1/n^c$, for any constant c (without changing the asymptotic complexity of the algorithm).

2.2 A Structural Ordering for α-Arboricity Graphs

Our analysis relies on a fixed acyclic orientation as described next.

▶ **Definition 4** (Ideal Peeling Layers). *Let $G = (V, E)$ be a graph, and let $G_0 = G$. For every integer $k \geq 1$, define recursively*

$$L_k^* := \{ v \in V(G_{k-1}) \mid d_{G_{k-1}}(v) \leq 3\alpha \},$$

and let

$$G_k := G\Big[V \setminus \bigcup_{i=1}^{k} L_i^*\Big]$$

be the subgraph induced by the vertices that remain after removing the first k layers.

For each vertex $v \in V$, define $\ell(v)$ to be the unique index k such that $v \in L_k^*$.

This peeling-based partition originates in the work of Barenboim and Elkin [2].

▶ **Lemma 5** (Structural Depth Bound). *The ideal peeling process terminates in $L = O(\log n)$ levels.*

Proof. Consider any subgraph $H \subseteq G$. Since G has arboricity α, $|E(H)| \leq \alpha|V(H)|$. By the Handshaking Lemma, $\sum_{v \in V(H)} d_H(v) \leq 2\alpha|V(H)|$. Let $S \subseteq V(H)$ be the set of vertices with $d_H(v) > 3\alpha$. Then $3\alpha|S| < \sum d_H(v) \leq 2\alpha|V(H)|$, implying $|S| < \frac{2}{3}|V(H)|$. Thus, each level removes at least $1/3$ of the remaining vertices, leading to $O(\log n)$ levels. ◀

▶ **Definition 6** (Good sample). *Fix a vertex v and a sampled neighbor set $S \subseteq N(v)$. We say that S is* good *(with respect to the peeling layers) if every vertex $u \in S$ belongs to a strictly lower level than v, i.e., $\ell(u) < \ell(v)$ for all $u \in S$.*

The structural levels are introduced purely for the purpose of analysis and are never accessed by the algorithm. Assume that each vertex initially possesses a good sample. Starting from a vertex v, we consider a recursive process that follows sampled (good) neighbors until it reaches a terminal vertex, defined as a vertex of degree at most 3α. Lemma 5 implies that the recursion depth of this process is bounded by $O(\log n)$. The depth of this recursion allows us to bound the number of iterations of our algorithm as we shall see next.

Our algorithm proceeds in $\Theta(\log n)$ iterations. In order to describe it we introduce several objects and definitions that will be used throughout the algorithm and its analysis.

Notation and Setup

For every vertex $v \in V$ and every iteration t, we use a different set of sampled neighbors. These samples are fixed implicitly by the algorithm's randomness and are used to guide the recursive exploration of the neighborhood of v at different paces during the level computation. We define these sets next.

2.2.1 Per-Iteration Samples

In this subsection, we describe how the samples associated with each vertex and iteration are generated. The parameters introduced here will be used throughout the paper.

Fix an arboricity α, an orientation target $r = x\alpha$ where $x = \Omega(\log n)$, and a bound on the number of iterations $T \stackrel{\text{def}}{=} \Theta(\log n)$.

For every vertex $v \in V$ and every iteration $t \in [T]$, the algorithm associates with v a collection of $\ell \stackrel{\text{def}}{=} \Theta(\log n)$ independent random neighbor samples

$$S_{v,1}^{(t)}, S_{v,2}^{(t)}, \ldots, S_{v,\ell}^{(t)} \subseteq N(v).$$

Each sample $S_{v,i}^{(t)}$ is generated by including every neighbor $u \in N(v)$ independently with probability $p \stackrel{\text{def}}{=} c \cdot \frac{\log n}{r}$, where c is a constant whose precise choice will be dictated by the inequalities appearing later in the proof.

Implementation details

Our analysis focuses on query complexity. By Lemma 2.3 of Dai, Ghaffari, and Portmann [5], the samples $S_{v,1}^{(t)}, \ldots, S_{v,\ell}^{(t)}$ can be generated in expected time proportional to their size. Consequently, the running time of our algorithm matches its query complexity up to constant factors.

The generation occurs according to the following schedule.

2.2.2 Per-Iteration Staggered Scheduling

Fix an iteration $t \in [T]$. We next describe the execution of iteration t of the algorithm which is essentially a staggered generation of the samples defined above.

Iteration t proceeds in steps $y = 1, 2, \ldots$ Let $i(y)$ denote the (unique) index of the bit that flips from 0 to 1 when incrementing $y - 1$ to y in binary. Equivalently, $i(y)$ is the index of the least significant 1 bit in the binary representation of y.

Scheduling rule

At time step y, the algorithm advances the generation of the sample $S_{v,i}^{(t)}$, where $i = i(y)$. Once the sample $S_{v,i}^{(t)}$ has been fully generated, the algorithm recursively processes its vertices using the same scheduling rule with the number of iterations decreased by 1. Thus, we obtain the following.

Execution frequency

The generation of $S_{v,i}^{(t)}$ and its recursion advance exactly at steps y satisfying

$$y \equiv 2^{i-1} \pmod{2^i}.$$

Therefore, over the first y steps, this advancement occurs $\Theta\left(\frac{y}{2^i}\right)$ times. Since each advancement accounts for 2^i underlying steps, the total processing time is obtained by multiplying the number of advancements by 2^i.

3 The algorithm for r-Orientation

Our LCA is organized into three procedures: Orient answers an edge query using the two endpoint levels; ComputeLevel tries iterations $t = 1, \ldots, T$ and returns the first successful one; and ComputeLevelIter implements a single iteration.

Algorithm 1 Orient(u, v): LCA for edge orientation.

Require: An undirected edge $\{u, v\}$
$a \leftarrow$ ComputeLevel(u)
$b \leftarrow$ ComputeLevel(v)
if $a < b$ **then**
    return $u \to v$
else if $b < a$ **then**
    return $v \to u$
else
    return orientation by vertex IDs
end if

The procedure ComputeLevel(v) determines the smallest iteration t in which v can be resolved within the probe budget by invoking ComputeLevelIter$(v, t, \cdot)$ for increasing t.

Algorithm 2 ComputeLevel(v).

Require: Vertex v
Ensure: $\ell(v) \in \{1, \ldots, L\} \cup \{\infty\}$
if $d(v) \leq 3\alpha$ **then**
    return 1
end if
for $t \leftarrow 1$ **to** T **do**
    $q \leftarrow 0$ ▷ local probes used in iteration t
    if ComputeLevelIter$(v, t, q) =$ **true then**
        return t
    end if
end for
return ∞

At iteration t, ComputeLevelIter explores sampled neighbors of v using the staggered schedule, and returns **true** iff it certifies that the sampled recursive exploration starting from v terminates without exceeding the local probe budget $B \stackrel{\text{def}}{=} \Theta(\alpha n \log n / r^2)$. More concretely, a sampled set is considered successful if all recursive calls made on vertices in that set return **true**. If a recursive call returns **false**, then this particular sampled set is discarded; however, the entire iteration does not necessarily fail, since the staggered schedule may still find another sampled set whose recursive calls all succeed. The iteration fails only if no sampled set succeeds before the budget is exceeded.

3.1 Analysis

In the analysis of our algorithm we use the following notion of processing time.

▶ **Definition 7** (Processing-time). *For a vertex v and an iteration $t \geq 1$, define* $\mathrm{Proc}_t(v)$ *recursively as*

$$\mathrm{Proc}_t(v) := \begin{cases} 1, & \text{if } d(v) \leq 3\alpha, \\ 1 + \min\limits_{i \in [\ell]} \left\{ 2^i \cdot \left(|S_{v,i}^{(t)}| + \sum\limits_{u \in S_{v,i}^{(t)}} \mathrm{Proc}_{t-1}(u) \right) \right\}, & \text{if } d(v) > 3\alpha. \end{cases}$$

and set $\mathrm{Proc}_0(v) := 1$ *for all* v.

Algorithm 3 ComputeLevelIter(v, t, q).

Require: Vertex v, iteration t, global counter q
Ensure: **true** iff iteration t succeeds for v

$q \leftarrow q + 1$
if $q > B$ **then**
  return false
end if
if $d(v) \leq 3\alpha$ **then**
  return true
end if
if $t = 0$ **then**
  return false
end if
Initialize the recursive processing of $S_{v,1}^{(t)}, \ldots, S_{v,\ell}^{(t)}$.
for each scheduling step $y = 1, 2, \ldots$ while $q \leq B$ **do**
  $i \leftarrow i(y)$
  Continue processing the vertices of $S_{v,i}^{(t)}$.
  For the next unprocessed vertex $u \in S_{v,i}^{(t)}$, invoke ComputeLevelIter$(u, t-1, q)$.
  if the recursive call returns **false then**
    Discard the sample $S_{v,i}^{(t)}$.
  else if all vertices of $S_{v,i}^{(t)}$ have been processed **then**
    return true
  end if
end for
return false

The factor 2^i appears because sample $S_{v,i}^{(t)}$ is advanced only once every 2^i scheduling steps. Thus, if the evaluation of this sample requires Q units of local work, then the global schedule spends $O(2^i Q)$ steps before this evaluation completes. This is why the processing time takes a minimum over terms of the form $2^i(\cdot)$.

By construction, $\mathrm{Proc}_t(v)$ upper bounds the number of probes used to evaluate iteration t rooted at v under the staggered schedule. For a fixed sample index i, recursively processing its sampled neighbors costs $\sum_{u \in S_{v,i}^{(t)}} \mathrm{Proc}_{t-1}(u)$, and the execution frequency contributes a multiplicative $\Theta(2^i)$ factor, giving the term $2^i(\cdot)$. Since the schedule runs all indices in parallel and stops when the first completes, we take the minimum over $i \in [\ell]$.

By Algorithm 2, the level of a vertex v is defined as follows.

▶ **Definition 8** (Level). *The level of a vertex v, $level(v)$ is defined to be the minimum $t \in T$ such that the processing time of* ComputeLevelIter$(v, t, \cdot)$ *is less than B, or ∞ if such t does not exist. In the latter case we call v and* overflow vertex.

3.1.1 Total expected work per iteration

In this subsection, we bound the expected total work per iteration, summed over all vertices, when excluding from the count the initial degree query performed for each vertex.

▷ Claim 9. For every $t \leq T$, $\mathbb{E}\left[\sum_{v \in V}(\mathrm{Proc}_t(v) - 1)\right] = O((n \log n)/x)$.

Proof. Recall that $p = c \cdot \frac{\log n}{r}$. Thus, for $r \geq 100c\alpha \log n$ it is bounded from above by $\frac{1}{100 \cdot \alpha}$. In each sample $S_{v,i}^{(t)}$, every neighbor of v is included independently with probability p.

Call a sample $S_{v,i}^{(t)}$ *good* if all sampled neighbors belong to strictly lower ideal peeling layers than v, i.e., $\ell(u) < \ell(v)$ for all $u \in S_{v,i}^{(t)}$ (Definitions 4 and 6). Since each vertex has at most 3α neighbors whose layer index is not strictly smaller than its own, we have

$$\Pr\left[S_{v,i}^{(t)} \text{ is good}\right] = (1-p)^{L_{\geq t}} \geq (1-p)^{3\alpha} \geq e^{-3/100} \geq 0.97.$$

Hence each sample is good with probability at least 0.97 and bad with probability at most 0.03. Let $i^\star$ be the index of the first good sample. Then

$$\Pr[i^\star = i] \leq (0.03)^{i-1} = q^{\,i-1}, \qquad q := 0.03.$$

Condition on the event that $i^\star = i$. By the definition of $\mathrm{Proc}_t(v)$, we have

$$\mathbb{E}[\mathrm{Proc}_t(v) - 1] \leq \sum_{i=1}^{\ell} \Pr[i^\star = i] \cdot 2^i \cdot \mathbb{E}\left[|S_{v,i}^{(t)}| + \sum_{u \in S_{v,i}^{(t)}} \mathrm{Proc}_{t-1}(u) \;\middle|\; i^\star = i \right].$$

Using the bound $\Pr[i^\star = i] \leq q^{\,i-1}$, this implies

$$\mathbb{E}[\mathrm{Proc}_t(v) - 1] \leq \sum_{i=1}^{\ell} q^{\,i-1} \cdot 2^i \, \mathbb{E}\left[\sum_{u \in S_{v,i}^{(t)}} |S_{v,i}^{(t)}| + \mathrm{Proc}_{t-1}(u) \;\middle|\; i^\star = i \right]. \tag{1}$$

Conditioned on the event that $S_{v,i}^{(t)}$ is a good sample, only neighbors u in strictly lower ideal layers than v can appear. Each such neighbor is included independently with probability p, so by linearity of expectation,

$$\mathbb{E}\left[|S_{v,i}^{(t)}| + \sum_{u \in S_{v,i}^{(t)}} \mathrm{Proc}_{t-1}(u) \;\middle|\; i^\star = i \right] = \sum_{\substack{u \in N(v) \\ \ell(u) < \ell(v)}} p \,(1 + \mathbb{E}[\mathrm{Proc}_{t-1}(u)]).$$

Plugging into (1) yields

$$\mathbb{E}[\mathrm{Proc}_t(v) - 1] \leq \sum_{i=1}^{\ell} q^{\,i-1} 2^i \sum_{\substack{u \in N(v) \\ \ell(u) < \ell(v)}} p\,(1 + \mathbb{E}[\mathrm{Proc}_{t-1}(u)]). \tag{2}$$

The geometric factors satisfy

$$\sum_{i=1}^{\ell} q^{\,i-1} 2^i \leq 2 \sum_{i=1}^{\infty} (2q)^{i-1} = \frac{2}{1-2q} = \frac{2}{1-0.06} < 3.$$

Therefore,

$$\mathbb{E}[\mathrm{Proc}_t(v) - 1] \leq 3 \sum_{\substack{u \in N(v) \\ \ell(u) < \ell(v)}} p\,(1 + \mathbb{E}[\mathrm{Proc}_{t-1}(u)]). \tag{3}$$

Summing (3) over all vertices v and rearranging gives

$$\sum_{v\in V}\mathbb{E}[\mathrm{Proc}_t(v)-1]\le 3p\sum_{u\in V}(1+\mathbb{E}[\mathrm{Proc}_{t-1}(u)])\cdot\left|v\in N(u):\ell(v)>\ell(u)\right|. \tag{4}$$

By the ideal peeling layers property, for every vertex u there are at most 3α neighbors v with $\ell(v)\ge\ell(u)$. Hence,

$$\sum_{v\in V}\mathbb{E}[\mathrm{Proc}_t(v)-1]\le 9\alpha p\sum_{u\in V}(1+\mathbb{E}[\mathrm{Proc}_{t-1}(u)])=9\alpha p\left(2|V|+\sum_{u\in V}\mathbb{E}[\mathrm{Proc}_{t-1}(u)-1]\right) \tag{5}$$

Let $R_t\stackrel{\text{def}}{=}\sum_{v\in V}\mathbb{E}[\mathrm{Proc}_t(v)-1]$. Then (5) implies

$$R_t\le 9\alpha p\left(2|V|+R_{t-1}\right).$$

Since $R_0=\sum_{v\in V}\mathbb{E}[\mathrm{Proc}_0(v)-1]=0$, and from the fact that $9\alpha p<0.5$ and that $9\alpha p=O((\log n)/x)$ it follows by induction that $R_t=O((n\log n)/x)$, as claimed ◁

▷ Claim 10. With high constant probability, the number of overflow vertices is at most $r/2$.

Proof. Observe that R_t is the expected total number of queries required to execute iteration t on *all* vertices, excluding only the first degree query performed on the starting vertex. By Claim 9,

$$\mathbb{E}\left[\sum_{v\in V}(\mathrm{Proc}_t(v)-1)\right]\le c_1(n\log n)/x,$$

where c_1 is a constant. Let

$$Z_t=\left|\left\{v\in V:\mathrm{Proc}_t(v)-1\ge c_2\cdot\frac{n\alpha\log n}{r^2}\right\}\right|,$$

where c_2 is a sufficiently large constant (the conditions it must satisfy will become apparent in the remainder of the proof). Then

$$Z_t\cdot c_2\frac{n\alpha}{r^2}\le\sum_{v\in V}(\mathrm{Proc}_t(v)-1).$$

Taking expectations,

$$\mathbb{E}[Z_t]\le\frac{c_1(n\log n)/x}{c_2\alpha n\log n/r^2}=O\left(\frac{c_1 r}{c_2}\right),$$

where we used $x=r/\alpha$. Choosing c_2 sufficiently large gives

$$\mathbb{E}[Z_t]\le\frac{r}{20}.$$

Therefore, by Markov's inequality,

$$\Pr\left[Z_t>\frac{r}{2}\right]\le\frac{\mathbb{E}[Z_t]}{r/2}\le\frac{1}{10}.$$

Thus, with high constant probability, at most $r/2$ vertices require at least $c_2\cdot\alpha n\log n/r^2$ additional queries, as claimed. ◁

3.2 Bounding the out-degree

▷ Claim 11. With high constant probability (over the randomness of the algorithm) the out-degree of every vertex is r.

Proof. Fix an iteration t, and consider only the randomness required by the algorithm for executing iterations $1, \ldots, t$, namely the sets $\{S_{v,i}^{(j)}\}_{j\in[t],\, i\in[\ell],\, v\in V}$. Let $U_{\leq t}$ denote the set of vertices that were unsuccessful in all iterations in $[t]$.

Let $v \in V$ be a vertex such that $|N(v) \cap U_{\leq t}| \geq r/2$. Fix an index $i \in [\ell]$. The probability that none of the vertices in $N(v) \cap U_{\leq t}$ belongs to $S_{v,i}^{(t)}$ is at most

$$(1-p)^{r/2} \leq e^{-pr/2} \leq 1/n^{c/2} \leq 1/n^2,$$

where the last inequality holds for an appropriate setting of c in the definition of p. By a union bound over all vertices v and all indices i, it follows that, with high probability, for every v and i, if $|N(v) \cap U_{\leq t}| \geq r/2$, then at least one vertex in $N(v) \cap U_{\leq t}$ belongs to $S_{v,i}^{(t+1)}$.

Consequently, it holds that $level(v) \geq t+1$ (note that a vertex v can be successful in iteration $t+1$ only if there exists $i \in [\ell]$ such that all vertices in $S_{v,i}^{(t+1)}$ were successful in one of the iterations in $[t]$ (this is a necessary but not a sufficient condition since the query budget is divided between the different procedures). We conclude that, with high probability, every vertex has at most $r/2$ neighbors whose level is greater than or equal to its own, excluding overflow vertices. Since w.h.c.p. the number of overflow vertices is bounded by $r/2$, this concludes the proof. ◁

We now prove Theorem 1.

Proof of Theorem 1. The claim on the query complexity follows from the construction as the probe complexity per edge query is $O\left(\frac{\alpha n \log^2 n}{r^2}\right)$, where the additional $\log n$ factor comes from trying $T = \Theta(\log n)$ iterations, and each iteration is stopped after budget $B = \Theta(\alpha n \log n / r^2)$. The claim on the success probability of the algorithm follows from Claim 11. ◀

References

1. Noga Alon, Ronitt Rubinfeld, Shai Vardi, and Ning Xie. Space-efficient local computation algorithms. In Yuval Rabani, editor, *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 1132–1139. SIAM, 2012. doi:10.1137/1.9781611973099.89.
2. Leonid Barenboim and Michael Elkin. Sublogarithmic distributed MIS algorithm for sparse graphs using nash-williams decomposition. *Distributed Comput.*, 22(5-6):363–379, 2010. doi:10.1007/S00446-009-0088-2.
3. Amartya Shankha Biswas, Talya Eden, Quanquan C. Liu, Ronitt Rubinfeld, and Slobodan Mitrovic. Massively parallel algorithms for small subgraph counting. In Amit Chakrabarti and Chaitanya Swamy, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2022, September 19-21, 2022, University of Illinois, Urbana-Champaign, USA (Virtual Conference)*, volume 245 of *LIPIcs*, pages 39:1–39:28. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.APPROX/RANDOM.2022.39.
4. Norishige Chiba and Takao Nishizeki. Arboricity and subgraph listing algorithms. *SIAM Journal on Computing*, 14(1):210–223, 1985. doi:10.1137/0214017.
5. Jiangqi Dai, Mohsen Ghaffari, and Julian Portmann. Constant approximation of arboricity in near-optimal sublinear time. In *Proceedings of the 2025 IEEE 66th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2025. doi:10.1109/FOCS63196.2025.00030.

6 Talya Eden, Saleet Mossel, and Dana Ron. Approximating the arboricity in sublinear time. In Joseph (Seffi) Naor and Niv Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9–12, 2022*, pages 2404–2425. SIAM, 2022. doi:10.1137/1.9781611977073.96.

7 Manuela Fischer, Mohsen Ghaffari, and Fabian Kuhn. Deterministic distributed edge-coloring via hypergraph maximal matching. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 180–191. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.25.

8 Pierre Fraigniaud, Marc Heinrich, and Adrian Kosowski. Local conflict coloring. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, Hyatt Regency, New Brunswick, New Jersey, USA, October 9-11, 2016*, pages 625–634. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.73.

9 Mohsen Ghaffari and Hsin-Hao Su. Distributed degree splitting, edge coloring, and orientations. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2505–2523. SIAM, 2017. doi:10.1137/1.9781611974782.166.

10 Meng He, Ganggui Tang, and Norbert Zeh. Orienting dynamic graphs, with applications to maximal matchings and adjacency queries. In Hee-Kap Ahn and Chan-Su Shin, editors, *Algorithms and Computation – 25th International Symposium, ISAAC 2014, Jeonju, Korea, December 15-17, 2014, Proceedings*, volume 8889 of *Lecture Notes in Computer Science*, pages 128–140. Springer, 2014. doi:10.1007/978-3-319-13075-0_11.

11 Akash Kumar, C. Seshadhri, and Andrew Stolman. Random walks and forbidden minors III: poly(d/ϵ)-time partition oracles for minor-free graph classes. In *62nd IEEE Annual Symposium on Foundations of Computer Science (FOCS 2021)*, pages 441–452. IEEE, 2021. doi:10.1109/FOCS52345.2021.00050.

12 Yishay Mansour and Shai Vardi. A local computation approximation scheme to maximum matching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2013)*, volume 8096 of *Lecture Notes in Computer Science*, pages 260–273. Springer, 2013. doi:10.1007/978-3-642-40328-6_19.

13 David W. Matula and Leland L. Beck. Smallest-last ordering and clustering and graph coloring algorithms. *Journal of Algorithms*, 3(3):187–200, 1983.

14 Slobodan Mitrović, Ronitt Rubinfeld, and Mihir Singhal. Locally Computing Edge Orientations. In *32nd Annual European Symposium on Algorithms (ESA 2024)*, volume 308 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 89:1–89:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ESA.2024.89.

15 Michal Parnas and Dana Ron. Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theor. Comput. Sci.*, 381(1-3):183–196, 2007. doi:10.1016/J.TCS.2007.04.040.

16 Ronitt Rubinfeld, Gil Tamir, Shai Vardi, and Ning Xie. Fast local computation algorithms. In Bernard Chazelle, editor, *Innovations in Computer Science – ICS 2011, Tsinghua University, Beijing, China, January 7-9, 2011. Proceedings*, pages 223–238. Tsinghua University Press, 2011. URL: http://conference.iiis.tsinghua.edu.cn/ICS2011/content/papers/36.html.

17 Shay Solomon and Nicole Wein. Improved dynamic graph coloring. *ACM Journal of Experimental Algorithmics*, 26(1), June 2020. doi:10.1145/3392724.

High Probability Streaming Lower Bounds for F_2 Estimation

William Swartworth
Carnegie Mellon University, Pittsburgh, PA, USA

David P. Woodruff
Carnegie Mellon University, Pittsburgh, PA, USA

Samson Zhou
Texas A&M University, College Station, TX, USA

Abstract

Estimating the second frequency moment (F_2) of an underlying frequency vector is a fundamental problem in the streaming model. While recent work by Braverman and Zamir [STOC 2025] resolved the space complexity for constant failure probability in the insertion-only model, the optimal dependence on the failure parameter δ remained open.

We close this gap by proving a tight high-probability lower bound of $\Omega\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\log\frac{\varepsilon\sqrt{n}}{\log(1/\delta)}\right)$ for $(1\pm\varepsilon)$-approximate F_2 estimation. The key challenge is the failure of prior multi-scale direct sum arguments under noise sensitivity. We introduce a noise-robust communication primitive, *Exam Mostly Set Disjointness*, and prove an $\Omega\left(\frac{m}{t}\log\frac{1}{\delta}\right)$ one-way lower bound. Embedding this into a multi-scale reduction yields the correct $\log(1/\delta)$ dependence.

We also give two complementary algorithms under natural structure assumptions. For streams with frequency bound B, we design a subsampling method using continuous F_0 tracking that replaces a $\log(n)$ factor with $\mathrm{polylog}(B)$. For k-sparse streams, we develop a two-stage sketch using approximate Morris counters, replacing $\log n$ with $\log k$ and achieving a further $\log\log m$ dependence on stream length.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases streaming algorithms, lower bounds, moment estimation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.76

Category RANDOM

Funding *David P. Woodruff*: David P. Woodruff is supported in part by Office of Naval Research award number N000142112647 and a Simons Investigator Award.
Samson Zhou: Samson Zhou is supported in part by NSF CCF-2335411 and gratefully acknowledges funding provided by the Oak Ridge Associated Universities (ORAU) Ralph E. Powe Junior Faculty Enhancement Award.

1 Introduction

The emergence of large-scale data applications has made streaming algorithms a fundamental framework for space-efficient computation. Such applications routinely generate massive datasets arising from sources such as social networks, web search systems, network monitoring, and distributed sensor platforms. A central problem in this model is the estimation of frequency moments. Here, elements from a universe $[n] := \{1, 2, \ldots, n\}$ arrive one by one in a stream, implicitly defining a frequency vector f, where $f_i \in \mathbb{N}$ denotes the number of occurrences of each element $i \in [n]$. We typically denote the stream length by m and assume that $n = \mathrm{poly}(m)$. The goal is to output, with high probability, a $(1\pm\varepsilon)$-approximation to


© William Swartworth, David P. Woodruff, and Samson Zhou;
licensed under Creative Commons License CC-BY 4.0

the p-th frequency moment of the stream, $F_p := \sum_{x \in U} f_x^p$, using space sublinear in both m and n. Since the seminal work of Alon, Matias, and Szegedy [2], the frequency moments problem has been closely studied for more than two decades [11, 3, 29, 18, 17, 25, 22, 21, 12, 9, 8, 6, 10, 13, 31, 32, 4, 19, 33, 15, 16, 26].

The second frequency moment ($p = 2$) is a fundamental quantity in data stream analysis. This statistic, commonly referred to as the repeat rate or surprise index, appears in a range of settings including database query optimization [1], approximate histogram maintenance [14], machine learning [28, 30], and network traffic anomaly detection [24], among others. In addition to these applications, F_2 provides a measure of skew or homogeneity in a dataset, which is important for performance optimization in database systems. In distributed and parallel computation, it is used to assess load imbalance and to guide how data should be partitioned across processors. It also informs other streaming tasks; for instance, some hybrid algorithms for estimating the number of distinct elements (F_0) rely on F_2 to decide which sampling strategy to apply. Exactly computing F_2 in a streaming setting requires storing all frequency counts, which leads to $\Omega(n)$ space usage and becomes impractical for large inputs. This has driven the development of streaming algorithms that maintain compact summaries of the data and produce accurate estimates in a single pass. Formally, the goal of the F_2 estimation problem is to compute a $(1 \pm \varepsilon)$-approximation with probability at least $1 - \delta$ while minimizing space. The classical AMS sketch of Alon, Matias, and Szegedy [2] achieves this using $O\left(\frac{1}{\varepsilon^2} \log \frac{1}{\delta} \log n\right)$ bits of space for a universe of size n. Despite extensive progress, the best known lower bound until 2025, due to Woodruff [29], was $\Omega\left(\frac{1}{\varepsilon^2} + \log n\right)$. Finally, a recent breakthrough by Braverman and Zamir [7] closed this gap for the case of constant failure probability δ in the insertion-only model, proving a tight lower bound of $\Omega(\frac{1}{\varepsilon^2} \log(n\varepsilon^2))$ space. When ε is not too small, e.g., $\varepsilon > n^{-1/2+c}$ for $c > 0$, this matches the AMS upper bound of $O\left(\frac{1}{\varepsilon^2} \log n\right)$. For smaller ε, [7] also provided a modified algorithm matching their lower bound. However, while the complexity in terms of n and ε was resolved, the optimal dependence on the failure probability δ remains an open question.

Our main result closes this gap by establishing a matching lower bound on the space complexity of F_2 estimation as a function of the failure probability δ, thereby resolving the dependence on δ in the insertion-only streaming model:

▶ **Theorem 1.** *Let $\mathcal{A}$ be a streaming algorithm that, for any data stream of length polynomial in n over a universe of size n, computes an estimate $\hat{F}_2$ such that $\Pr[|\hat{F}_2 - F_2| \le \varepsilon F_2] \ge 1 - \delta$. For $\varepsilon\sqrt{n} \ge \log \frac{1}{\delta}$, the space used by $\mathcal{A}$ is at least $\Omega\left(\log\left(\frac{\varepsilon\sqrt{n}}{\log \frac{1}{\delta}}\right) \frac{1}{\varepsilon^2} \log\left(\frac{1}{\delta}\right)\right)$.*

Beyond the main lower bound, our techniques yield additional results at both the communication and algorithmic levels. At the core of our approach is a new communication primitive called Exam Mostly Set Disjointness (EMostlyDISJ). In this problem, t players hold sets with a total size parameter m, and a referee holds a target element. The players' sets are promised to be either κ-almost disjoint (meaning the total number of repeated occurrences of elements across the sets is at most κ) or to have a unique element common to at least a constant fraction of the sets. The referee must decide whether the latter case holds and the common element matches their target.

▶ **Theorem 2** (EMostlyDISJ lower bound)**.** *Any one-way communication protocol for the t-player EMostlyDISJ problem with failure probability at most δ requires $\Omega\left(\frac{m}{t} \log \frac{1}{\delta}\right)$ communication when the almost-disjointness parameter is $\kappa = O\left(\log \frac{1}{\delta}\right)$. (Informal, see Corollary 24.)*

We then use this lower bound as a core primitive in our reduction framework, together with a multi-scale embedding argument, to obtain the streaming lower bound above. On the algorithmic side, we identify structured regimes where faster algorithms are possible.

▶ **Theorem 3** (Bounded-frequency streams). *Let $x \in \mathbb{Z}^n_{\geq 0}$ be the frequency vector of an insertion-only stream of length m. Suppose that $0 \leq x_i \leq B$ for all $i \in [n]$. Then there exists a streaming algorithm that, with probability at least $1-\delta$, computes a $(1\pm\varepsilon)$-approximation to $\|x\|_2^2$ using space (in bits) $O\left(\frac{1}{\varepsilon^2}\log^2\frac{B}{\varepsilon}\log\frac{1}{\delta}\left(\log B+\log\frac{1}{\varepsilon}\right)\right)+O\left(\log\frac{n}{\delta}\right)$.*

▶ **Theorem 4** (Sparse streams). *Let $x \in \mathbb{Z}^n_{\geq 0}$ be the frequency vector of an insertion-only stream of length m. Suppose that $\|x\|_0 \leq k$. Then there exists a streaming algorithm that, with probability at least $1-\delta$, computes a $(1\pm\varepsilon)$-approximation to $\|x\|_2^2$ using space (in bits) $O\left(\frac{1}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\left(\log\frac{k}{\varepsilon}+\log\log m\right)+\log n\cdot\log\frac{1}{\delta}\right)$.*

Together, these results provide a refined picture of the complexity of F_2 estimation, capturing both its limitations in the general case and improved guarantees under natural structural assumptions.

2 Background and Technical Overview

Proving strong streaming space lower bounds via communication complexity often encounters a known barrier: traditional reductions struggle to amplify the hardness of a single communication problem to achieve the logarithmic factors (e.g., $\log n$) required for F_2 estimation. A recent breakthrough by Braverman and Zamir [7] overcame this barrier by introducing a novel multi-scale direct sum argument applied directly at the streaming level, built on top of a specialized communication model.

The Braverman-Zamir Approach and Exam Set Disjointness. At the core of their lower bound is the *Exam Set Disjointness* (EDISJ) problem. In this communication game, t players each hold a set of items, and a referee holds a single target element y. The players communicate sequentially, eventually sending a message to the referee, who must determine if y is the *unique* intersection of all the players' sets. This "exam" model is crucial because it forces the players to communicate enough information to verify a specific intersection, rather than merely detecting whether any intersection exists. They showed that solving EDISJ requires $\Omega(m/t)$ communication, where m is the total size of the sets.

To lift this communication bound to the streaming setting, Braverman and Zamir embed multiple instances of EDISJ into a single stream. These instances encode games with varying numbers of players (e.g., $2, 4, 8, \ldots$, up to roughly $\varepsilon\sqrt{n}$) and are interleaved at different "scales" or "levels." Because these instances share parts of the stream, they are dependent, and standard direct sum theorems do not apply. Instead, they developed a clever information-theoretic argument to show that a streaming algorithm must effectively solve all these dependent instances simultaneously. At any arbitrary point j in the stream, an algorithm needs information about the "past" relevant to each scale. For a level l (corresponding to 2^l players), the algorithm must remember information about the preceding $n/2^l$ elements of the stream to solve the EDISJ instance at that scale. They demonstrated that these different-scaled pasts are sufficiently disjoint that the required information adds up. Summing the $\Omega(1/\varepsilon^2)$ cost across each of the $\Theta(\log(n\varepsilon^2))$ scales yields their final $\Omega(\frac{1}{\varepsilon^2}\log n)$ space bound.

The Barrier to Optimal Error Dependence. While this multi-scale argument elegantly resolves the space complexity in terms of n and ε, it fails to capture the optimal dependence on the failure probability δ. The limitation stems directly from the EDISJ problem itself: the standard Set Disjointness problem, and its exam variant, are highly sensitive to noise. Because

the YES instance relies on a single, perfect intersection, an algorithm that is permitted to fail with probability δ might simply miss this unique intersection without paying the full information cost. Consequently, EDISJ cannot yield the robust $\log(1/\delta)$ communication lower bounds necessary for stronger high-probability streaming guarantees.

Our Contributions: Exam Mostly Set Disjointness. To achieve tight dependence on δ, we adapt the Braverman-Zamir framework by replacing EDISJ with a new, noise-robust communication primitive that we call *Exam Mostly Set Disjointness* (EMostlyDISJ). This problem is a synthesis of the Exam Set Disjointness problem from [7] and the Mostly Set Disjointness problem introduced by Kamath et al. [20].

Instead of demanding perfect disjointness, EMostlyDISJ relaxes the conditions to make the problem strictly harder for algorithms prone to error. By distinguishing between sets being almost disjoint (few intersections) and having an element common to a large fraction of the sets, the problem becomes robust to failure probabilities.

▶ **Definition 5** (Exam Mostly Set Disjointness (EMostlyDISJ)). *The setup involves t players and one referee. Let U be a universe of size $|U|$. The inputs are sets $S_1, \ldots, S_t \subseteq U$ for the players and an element $j \in U$ for the referee. The input sets are promised to be either (i) κ-almost disjoint (the total number of repeated occurrences of elements across the sets is at most κ) or (ii) to have a unique element $j_0 \in U$ that is common to at least ct of the sets for some constant $c \in (0, 1)$, and the remaining elements across the sets are κ-almost-disjoint. The communication is one-way from player i to player $i + 1$, and finally to the referee. With failure probability at most δ, the referee must decide if the input is an instance of case (ii) and the intersecting element j_0 is equal to its element j.*

We prove that this game requires $\Omega\left(\frac{m}{t}\log\frac{1}{\delta}\right)$ communication when the almost-disjointness parameter is bounded by $M = O\left(\log\frac{1}{\delta}\right)$.

Putting it Together: The Multi-Scale Reduction. With the robust communication bound in hand, we plug EMostlyDISJ directly into the multi-scale direct sum framework. In our reduction, the stream is divided into blocks corresponding to players, and elements are grouped into "super-items" of size $d = \Theta(\frac{\varepsilon^2 n}{t^2})$. If a super-item is repeated among a constant fraction of the players (the "Yes" case of EMostlyDISJ), the F_2 moment of the stream increases significantly compared to the "No" case, where items are repeated at most M times. By embedding these robust EMostlyDISJ instances across multiple scales (different values of t) and applying the direct sum argument over the disjoint "pasts," the robust $\log(1/\delta)$ communication costs naturally accumulate at every valid level of the stream, culminating in our complete and tight space lower bound for F_2 estimation.

Bounded-frequency streams. We consider two classes of insertion-only streams for which the general space lower bound above does not apply. By exploiting structural assumptions about the data, we show that it is possible to beat the lower bound with more space-efficient algorithms. The first class is streams with entries bounded by B, i.e., the maximum frequency of any element is $x_i \leq B$. The main observation is that when B is small, we obtain a good estimate of $F_2(x)$ by subsampling entries from the support of x. We implement this by applying a pairwise independent hash function to filter the stream down to an expected effective dimension of roughly $K = O(\frac{B^2}{\varepsilon^2})$. Because the frequencies are strictly bounded by B, the variance of this sampling-based estimator remains well-controlled. This subsampling reduces the size of the vector dramatically. When we apply the classical AMS sketch of [2]

purely to this restricted substream, the counters accumulate much smaller values (bounded by $O(BK)$). This drastically lowers the bit complexity per entry, effectively reducing the dominant multiplicative space dependence on $\log n$ down to roughly $\log B$.

There are several complications in following the outlined procedure in a single-pass streaming model. In particular, the optimal subsampling rate needs to depend inversely on the total support size (F_0), which we do not have access to at the start of the stream and grows dynamically. To handle this, we use a continuous F_0 tracker to estimate the sparsity at all points in time. As the stream progresses and our F_0 estimate crosses power-of-two thresholds, we dynamically spawn new sets of sketches tuned to progressively smaller sampling rates. By maintaining only a small active window of parallel instances (specifically, $O(\log(B/\varepsilon))$ instances) at any given time and discarding older ones, we successfully isolate a sketch with the correct sampling scale for the suffix of the stream, while bounding the error accumulated from the prefix.

Sparse streams. We also consider sparse streams, where the underlying frequency vector has at most k non-zero entries (i.e., $\|x\|_0 \leq k$). For this setting, our strategy utilizes a two-stage dimensionality reduction tailored specifically for sparse vectors. First, we hash the massive n-dimensional universe items into $M = \Theta(k/\varepsilon^2)$ random buckets. Because the underlying vector is guaranteed to be k-sparse, mapping the non-zero elements into this number of buckets results in very few collisions. This ensures that the F_2 moment of the resulting compressed, low-dimensional vector closely preserves the original F_2 moment up to a $(1 \pm \varepsilon/4)$ factor. We then compose this with an AMS sketch applied directly to the resulting compressed vector. To aggressively optimize the space and further remove lingering logarithmic dependencies on the stream length m, we do not store the entries of the AMS sketch exactly. The AMS sketch relies on linear combinations with random ± 1 signs, so we conceptually decompose each sketch counter into separate positive and negative updates. We then use Morris Counters [27] to separately estimate the positive and negative contributions to each bucket. The key to the error analysis relies on the structure of sparse vectors: by the Cauchy-Schwarz inequality, any k-sparse stream of length m satisfies $F_2 \geq m^2/k$. By tuning the Morris counters to operate with an error parameter of $\varepsilon' = O(\varepsilon/k)$, the additive error introduced by the approximate counting is tightly bounded, ensuring it is small enough to be absorbed into the overall $(1 \pm \varepsilon)$ multiplicative error. This layered approach roughly allows us to replace the heavy $\log n$ dependence with $\log k$ (alongside a deeply sub-logarithmic $\log \log m$ factor), where k is the sparsity.

3 Preliminaries

In this section, we formally define the fundamental concepts, models, and tools used throughout the paper. We review key concepts from information theory and communication complexity, and briefly describe the standard probability bounds and sketching algorithms utilized in our upper bounds.

3.1 Information Theory

Our space lower bounds rely heavily on information-theoretic measures to quantify how much the internal memory of a streaming algorithm "knows" about the input stream. We use standard concepts from Shannon information theory. All logarithms are taken to base 2. Let X, Y, and Z be discrete random variables.

▶ **Definition 6** (Entropy and Conditional Entropy). *The Shannon entropy of X, which measures its expected uncertainty or information content, is defined as $H(X) := \sum_{x \in \mathrm{supp}(X)} \Pr[X = x] \log\left(\frac{1}{\Pr[X=x]}\right)$. The conditional entropy of X given Y is the expected remaining uncertainty of X after Y is observed, $H(X|Y) := \sum_{y \in \mathrm{supp}(Y)} \Pr[Y = y] H(X \mid Y = y) = H(X,Y) - H(Y)$. A fundamental property of entropy is that conditioning cannot increase uncertainty: $H(X|Y) \leq H(X)$, with equality holding if and only if X and Y are completely independent.*

▶ **Definition 7** (Mutual Information). *The mutual information between X and Y quantifies the amount of information that observing Y reveals about X (and vice versa), $I(X;Y) := H(X) - H(X|Y) = H(Y) - H(Y|X)$. By definition, mutual information is symmetric and non-negative: $I(X;Y) \geq 0$. The mutual information between X and Y conditioned on a third variable Z is defined as $I(X;Y \mid Z) := H(X|Z) - H(X|Y,Z)$.*

▶ **Lemma 8** (Chain Rules). *Information-theoretic quantities can be decomposed over sequences of random variables using chain rules. For a sequence of random variables $X_1, X_2, \ldots, X_k$, and letting $X_{<i}$ denote the prefix $(X_1, \ldots, X_{i-1})$:*

- ***Chain Rule for Entropy:*** $H(X_1, \ldots, X_k \mid Y) = \sum_{i=1}^{k} H(X_i \mid X_{<i}, Y)$.
- ***Chain Rule for Mutual Information:*** $I(X_1, \ldots, X_k; Y \mid Z) = \sum_{i=1}^{k} I(X_i; Y \mid X_{<i}, Z)$.

▶ **Lemma 9** (Data Processing Inequality). *If X, Y, and Z form a Markov chain $X \to Y \to Z$ (meaning Z is conditionally independent of X given Y), then algorithmic post-processing cannot generate new information: $I(X;Z) \leq I(X;Y)$.*

Because a streaming algorithm's memory state is a function of the stream prefix and internal randomness, the data processing inequality is commonly used to bound the information the algorithm retains about the original input stream.

3.2 Communication Complexity and Information Cost

To lower bound the space complexity of a streaming algorithm, we rely on reductions from multi-party communication games.

▶ **Definition 10** (The Multi-Party One-Way Model). *In the t-player one-way communication model, t players $P_1, P_2, \ldots, P_t$ receive respective inputs $X_1, X_2, \ldots, X_t$ drawn from a joint probability distribution μ. A referee may also receive an independent input y. The players collaborate to compute a joint function $f(X_1, \ldots, X_t, y)$ or solve a decision problem.*

The communication flows strictly one-way: player P_1 computes a message based on their input X_1 and sends it to P_2. Then, P_2 computes a message based on X_2 and the message from P_1, sending it to P_3, and so on. The final player P_t sends a message to the referee, who must output the correct answer.

▶ **Definition 11** (Protocols and Transcripts). *A randomized protocol Π governs the players' message generation. Players may utilize local private randomness as well as a shared* public randomness *string R, which is visible to all participants without incurring communication costs. The* transcript *of the protocol, denoted $\Pi(X)$ (or simply Π), is the concatenation of all messages sent over the channel. The* communication cost *of Π is the worst-case bit-length of its transcript.*

▶ **Definition 12** (Conditional Information Cost). *While traditional communication complexity bounds the raw bit-length of the transcript,* information complexity *measures the actual amount of information the transcript reveals about the inputs. For a protocol* Π *running on inputs* $X \sim \mu$ *and utilizing public randomness* R*, the* conditional information cost *is defined as the mutual information between the inputs and the transcript, conditioned on the public randomness,* $I_\mu(X_1, \ldots, X_t; \Pi \mid R)$*. By Shannon's source coding theorem, the expected communication cost of a protocol is always lower-bounded by its information cost:* $\mathbb{E}[|\Pi|] \geq I_\mu(X; \Pi \mid R)$.

Reduction to streaming space. A streaming algorithm $\mathcal{A}$ evaluated on a stream naturally induces a one-way communication protocol. By partitioning the stream into t contiguous blocks assigned to t players, Player i simulates $\mathcal{A}$ on their block and passes the resulting memory state $\mathcal{M}$ as their message to Player $i+1$. Thus, proving a lower bound on the information cost of a distributed game strictly lower bounds the space complexity of any streaming algorithm capable of solving that game.

3.3 Algorithmic Primitives and Probabilistic Tools

In our upper bound constructions, we utilize the following standard sketching tools and probability structures:

▶ **Lemma 13** (Concentration Inequalities). *We frequently apply standard probability tail bounds to control algorithm failure rates:*

- ***Markov's Inequality:*** *For any non-negative random variable* X *and* $a > 0$, $\Pr[X \geq a] \leq \frac{\mathbb{E}[X]}{a}$.
- ***Chernoff Bound:*** *For a sum of independent indicator variables* $X = \sum_{i=1}^{k} X_i$*, the probability of a large relative deviation is bounded exponentially. For any* $\delta > 0$, $\Pr[X \geq (1+\delta)\mathbb{E}[X]] \leq \exp\left(-\frac{\delta^2 \mathbb{E}[X]}{2+\delta}\right)$ *and* $\Pr[X \leq (1-\delta)\mathbb{E}[X]] \leq \exp\left(-\frac{\delta^2 \mathbb{E}[X]}{2}\right)$.

▶ **Definition 14** (k-wise Independent Hash Functions). *A family of hash functions* $\mathcal{H} = \{h : [n] \to [M]\}$ *is* k-wise independent *if, for any* k *distinct elements* $x_1, \ldots, x_k \in [n]$ *and any target values* $y_1, \ldots, y_k \in [M]$*, a function* h *chosen uniformly at random from* $\mathcal{H}$ *satisfies* $\Pr[h(x_1) = y_1 \wedge \ldots \wedge h(x_k) = y_k] = \frac{1}{M^k}$*. Storing such a function requires only* $O(k \log(\max(n, M)))$ *bits, which provides highly memory-efficient pseudo-randomness. We utilize pairwise (*2*-wise) and* 4*-wise independent hash functions in our sketches.*

▶ **Theorem 15** (AMS Sketch). *[2] There exists a streaming algorithm that uses* $O\left(\frac{1}{\varepsilon^2} \log n\right)$ *bits of space and outputs a* $(1+\varepsilon)$*-approximation to the* F_2 *moment with constant probability.*

We remark that the AMS algorithm utilizes a 4-wise independent hash function $\sigma : [n] \to \{-1, 1\}$ and maintains a linear counter $Z = \sum_{j=1}^{m} \sigma(a_j) = \sum_{i=1}^{n} \sigma(i) f_i$, whose square Z^2 serves as an unbiased estimator for F_2 with bounded variance.

▶ **Theorem 16** (Morris Counters). *[27] There exists a streaming algorithm that outputs a constant-factor approximation to the count of size* n *with constant probability, using* $O(\log \log n)$ *bits of space.*

To track a count up to N, a Morris counter explicitly stores a logarithmic value c (requiring only $O(\log \log N)$ bits). To update the counter, c is incremented with probability 2^{-c}, giving an unbiased estimator for the true count.

▶ **Theorem 17** (F_0 Estimators). *[23, 5] There exists a streaming algorithm that uses $O\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}+\log n\right)$ bits of space and with constant probability, outputs a $(1+\varepsilon)$-approximation to the F_0 moment at all points in the stream.*

4 Insertion-Only Lower Bounds for F_2 Estimation

We present a strengthened space lower bound for the problem of F_2 estimation in the insertion-only model, incorporating the optimal dependence on the failure probability δ. The proof strategy closely follows the multi-scale framework introduced by [7], but replaces their underlying communication problem with a robust variant that accounts for error probability.

▶ **Theorem 1.** *Let $\mathcal{A}$ be a streaming algorithm that, for any data stream of length polynomial in n over a universe of size n, computes an estimate $\hat{F}_2$ such that $\Pr[|\hat{F}_2 - F_2| \le \varepsilon F_2] \ge 1-\delta$. For $\varepsilon\sqrt{n} \ge \log\frac{1}{\delta}$, the space used by $\mathcal{A}$ is at least $\Omega\left(\log\left(\frac{\varepsilon\sqrt{n}}{\log\frac{1}{\delta}}\right)\frac{1}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right)$.*

We utilize standard concepts from information theory. For a communication protocol Π with inputs X drawn from a distribution μ and public randomness R, the conditional information cost is the mutual information $I(X;\Pi(X)|R)$ between the input and the protocol transcript, conditioned on the public randomness. The communication cost of any protocol is an upper bound on its information cost. For a streaming algorithm processing a random stream X, we analyze the information $I(X;\mathcal{M})$ that the memory state $\mathcal{M}$ contains about the stream.

4.1 The Exam Mostly Set Disjointness Problem

The core of the streaming space lower bound in [7] relies on a communication game called "Exam Set Disjointness" (EDISJ). However, EDISJ is highly sensitive to noise; a protocol can fail to identify a unique intersecting element by simply making an error on a single critical coordinate. Consequently, it cannot yield the robust $\Omega(\log(1/\delta))$ information cost bounds required to establish optimal dependencies on the failure probability δ. To overcome this limitation, we introduce a noise-robust variant inspired by the "Mostly Set Disjointness" problem introduced by Kamath et al. [20]. We first define a technical relaxation of disjointness that permits a bounded amount of noise in the sets.

▶ **Definition 18** (κ-Almost-Disjointness). *A collection of sets $S_1,\dots,S_t \subseteq U$ is said to be κ-almost-disjoint if the total number of non-unique element memberships is at most κ. Formally, the number of pairs (i,x) where x appears in S_i and at least one other set S_j is bounded by κ, $\left|\{(i,x)\in[t]\times U : x\in S_i \text{ and } x\in S_j \text{ for some } j\neq i\}\right| \le \kappa$.*

With this relaxation, we define our core communication primitive.

▶ **Definition 19** (Exam Mostly Set Disjointness (EMostlyDISJ)). *The setup involves t players and one referee. Let U be a universe of items. The inputs are sets $S_1,\dots,S_t \subseteq U$ for the players and an element $j\in U$ for the referee. The input sets are promised to satisfy exactly one of two conditions:*

- ***NO Instance:*** *The sets $S_1,\dots,S_t$ are globally κ-almost-disjoint.*
- ***YES Instance:*** *There is a unique element $j_0\in U$ that is common to at least ct of the sets for some constant $c\in(0,1)$, and the remaining elements across the sets are κ-almost-disjoint.*

The communication is one-way from player 1 to player t, and finally to the referee. With failure probability at most δ, the referee must decide if the input is a YES instance and the heavily intersecting element j_0 is exactly equal to the referee's target element j.

4.1.1 The One-Bit Problem: $F_{ct,t}$

To establish the information complexity of EMostlyDISJ, we reduce it to a primitive one-bit problem, much like how standard Set Disjointness reduces to the multi-party AND function.

In the $F_{ct,t}$ problem, each of the t players receives a single bit $Y_i \in \{0,1\}$. Their goal is to distinguish between two cases regarding the Hamming weight of the joint input vector $Y = (Y_1, \ldots, Y_t)$: a NO instance where the Hamming weight is at most 1, and a YES instance where the Hamming weight is at least ct.

We analyze the information cost of this problem with respect to a hard distribution ν over NO instances, adapting the approach of [20]. By bounding the information revealed on NO instances, we can embed many instances of this problem into a continuous stream.

▶ **Definition 20** (Hard Distribution ν). *Let R be a public random variable drawn uniformly from the player indices $[t] = \{1, 2, \ldots, t\}$. The distribution ν over the joint inputs $Y \in \{0,1\}^t$ is defined conditionally based on R: if $R = i$, the input Y is the all-zeros vector 0^t with probability $1/2$, and the standard basis vector e_i (where only the i-th coordinate is 1) with probability $1/2$.*

▶ **Lemma 21** (Conditional Information Cost on ν, adapted from [20]). *Let Π be a one-way communication protocol that solves $F_{ct,t}$ with failure probability at most δ. Let the inputs Y be drawn from the distribution ν. The conditional information cost of the protocol with respect to ν is lower-bounded by $I_\nu(Y; \Pi \mid R) \geq \Omega\left(\min\left(1, \frac{1}{t}\log\left(\frac{1}{\delta}\right)\right)\right)$.*

Proof. Assume without loss of generality that $\delta \leq 1/4$. Lemma 3.7 of [20] establishes a constant lower bound for protocols with sufficiently small error. Specifically, if a protocol has a failure probability δ' that satisfies $t \leq c' \log(1/(2e\delta'))$ for some absolute constant c', then its conditional information cost is bounded below by a constant $I_\nu(Y; \Pi \mid R, C) \geq \Omega(1)$, where C represents the public randomness. If the base protocol's failure probability δ already satisfies this condition ($t \leq c' \log(1/(2e\delta))$), then no amplification is needed. The lower bound is $\Omega(1)$ directly. To handle arbitrary failure probabilities δ, we use an information-theoretic amplification argument. Suppose we have a base protocol Π that fails with probability δ. We construct a boosted protocol Π' by executing Π independently r times using independent private randomness and independent public coins $C_1, \ldots, C_r$, and having the referee output the majority vote of the results. By a standard Chernoff bound, the failure probability of the boosted protocol Π' is bounded by $\delta' \leq (4\delta)^{r/2}$.

We choose the number of repetitions to be $r = \Theta\left(\frac{t}{\log(1/\delta)}\right)$. This ensures that the amplified error δ' is driven down precisely to the threshold required by the condition from [20]. The transcript of the boosted protocol Π' simply consists of r independent transcripts from the base protocol, denoted $(\Pi^1, \ldots, \Pi^r)$. By applying the established lower bound for the small-error regime, we know the information cost of Π' is bounded below by a constant $\Omega(1) \leq I_\nu(\Pi^1, \ldots, \Pi^r; Y \mid R)$. We now decompose this joint information using the chain rule for mutual information, $\Omega(1) \leq \sum_{i=1}^{r} I_\nu(\Pi^i; Y \mid R, C_1, \ldots, C_r, \Pi^{<i})$, where $\Pi^{<i}$ denotes the sequence of transcripts from the first $i-1$ repetitions. By an averaging argument, there must exist at least one index $i \in \{1, \ldots, r\}$ that contributes at least the average amount of information to the sum, $I_\nu(\Pi^i; Y \mid R, C_1, \ldots, C_r, \Pi^{<i}) \geq \frac{\Omega(1)}{r} = \Omega\left(\frac{1}{t}\log\frac{1}{\delta}\right)$. Finally, we decouple this term from the transcripts of the previous repetitions. By expanding the conditional mutual information into Shannon entropies, we have:

$$\begin{aligned} I_\nu(\Pi^i; Y \mid R, C_1, \ldots, C_r, \Pi^{<i}) &= H(\Pi^i \mid R, C_1, \ldots, C_r, \Pi^{<i}) - H(\Pi^i \mid R, C_1, \ldots, C_r, \Pi^{<i}, Y) \\ &\leq H(\Pi^i \mid R, C_i, \Pi^{<i}) - H(\Pi^i \mid R, C_i, Y). \end{aligned}$$

The last inequality holds because the transcripts of separate repetitions are generated using independent private randomness and public coins; thus, conditioned on the exact inputs Y, public randomness C_i and the selector R, Π^i is entirely independent of the previous transcripts $\Pi^{<i}$ and other public coins. Next, we use the fundamental property that conditioning reduces entropy, which implies $H(\Pi^i \mid R, C_i, \Pi^{<i}) \leq H(\Pi^i \mid R, C_i)$. Substituting this upper bound into our equation yields $I_\nu(\Pi^i; Y \mid R, C_1, \ldots, C_r, \Pi^{<i}) \leq H(\Pi^i \mid R, C_i) - H(\Pi^i \mid R, C_i, Y) = I_\nu(\Pi^i; Y \mid R, C_i)$. Because Π^i is identical in distribution to a single invocation of our original protocol Π utilizing public coins C, we conclude that the conditional information cost of the base protocol itself satisfies $I_\nu(\Pi; Y \mid R, C) \geq \Omega\left(\frac{1}{t} \log \frac{1}{\delta}\right)$. This completes the proof. ◀

4.1.2 From Hard Distribution to Product Distribution

To embed instances of the EMostlyDISJ communication game into a continuous streaming algorithm lower bound, we must transition from the artificially correlated hard distribution ν to a fully independent product distribution. We define μ_p as the product distribution over $\{0,1\}^t$ where each bit Y_i is set to 1 independently with probability p. Because data stream elements appear independently, simulating the stream naturally generates inputs drawn from μ_p. The following lemma, adapting a crucial technique from [7], establishes that the information cost lower bound holds even under this independent product distribution, scaling proportionately with p.

▶ **Lemma 22** (Adapted from Lemma 4.4 of [7]). *Let Π be a communication protocol for $F_{ct,t}$ with failure probability at most δ. Let $p < 1/t$. For the independent product distribution μ_p, the information cost of the protocol is lower-bounded by $I_{\mu_p}(Y; \Pi) \geq \Omega\left(p \cdot \min\left(t, \log\left(\frac{1}{\delta}\right)\right)\right)$.*

Proof. We shall define a probabilistic coupling between the product distribution μ_p and the hard distribution ν. We achieve this by introducing an auxiliary indicator variable $D \in \{0,1\}$ that acts as a filter, where conditioning on $D = 1$ forces $Y \sim \mu_p$ to perfectly emulate $Y \sim \nu$.

Step 1: Bounding probabilities under μ_p. First, we evaluate the probabilities of the critical NO instances (the all-zeros vector 0^t and the standard basis vectors e_i) under the product distribution μ_p. Since $p < 1/t$, the probability of observing the all-zeros vector is bounded below by a constant $\mu_p(0^t) = (1-p)^t \geq \left(1 - \frac{1}{t}\right)^t \geq \frac{1}{4}$ (for $t \geq 2$). Similarly, the probability of observing any specific standard basis vector e_i is $\mu_p(e_i) = p(1-p)^{t-1} \geq p\left(1 - \frac{1}{t}\right)^{t-1} \geq \frac{p}{e} \geq \frac{p}{4}$.

Step 2: Constructing the coupling variable D. Recall that under the marginalized hard distribution ν, the probabilities are $\nu(0^t) = 1/2$ and $\nu(e_i) = 1/(2t)$. Because the probabilities of these events under μ_p strictly dominate their relative frequencies under ν (up to a scaling factor), we define a boolean random variable D as a function of $Y \sim \mu_p$ so that:
1. Conditioned on $D = 1$, the distribution of Y is exactly the marginal distribution of ν.
2. The probability of the coupling succeeding is maximized: $\Pr[D = 1] = \Theta(p \cdot t)$.
By Bayes' theorem, ensuring $(Y \mid D = 1) \sim \nu$ requires setting the conditional probabilities as $\Pr[D = 1 \mid Y = y] = \frac{\nu(y) \Pr[D=1]}{\mu_p(y)}$. This is a valid probability assignment as long as the ratio on the right-hand side never exceeds 1. For $y = 0^t$, the ratio $\nu(0^t)/\mu_p(0^t)$ is upper bounded by $(1/2)/(1/4) = 2$. For $y = e_i$, the ratio $\nu(e_i)/\mu_p(e_i)$ is upper bounded by $(1/(2t))/(p/4) = 2/(pt)$. Thus by setting $\Pr[D = 1] = pt/2$, it follows that all conditional probabilities are contained in $[0,1]$ (since $p < 1/t$, the probability for 0^t evaluates to $pt < 1$, and for e_i evaluates to a constant), confirming that $\Pr[D = 1] = \Theta(p \cdot t)$. For all other vectors $y \notin \{0^t, e_1, \ldots, e_t\}$, we simply set $\Pr[D = 1 \mid Y = y] = 0$.

Because D is generated solely based on the realized value of Y, the variables form a Markov chain $D \to Y \to \Pi$. This implies that, conditioned on Y, the protocol transcript Π is entirely independent of D, yielding $I_{\mu_p}(D; \Pi \mid Y, C) = 0$.

Step 3: Chain rule expansion for D. We now use the chain rule for mutual information to relate the information cost under μ_p to the cost conditioned on the coupling success. We expand the joint mutual information $I_{\mu_p}(Y, D; \Pi)$ in two different ways:

$$\begin{aligned} I_{\mu_p}(Y; \Pi \mid C) &= I_{\mu_p}(Y, D; \Pi \mid C) - I_{\mu_p}(D; \Pi \mid Y, C) \\ &= I_{\mu_p}(Y, D; \Pi \mid C) \quad \text{(by the Markov property)} \\ &= I_{\mu_p}(D; \Pi \mid C) + I_{\mu_p}(Y; \Pi \mid D, C) \end{aligned}$$

By the non-negativity of mutual information, $I_{\mu_p}(D; \Pi \mid C) \geq 0$. Thus, we can expand the conditional mutual information over the states of D:

$$\begin{aligned} I_{\mu_p}(Y; \Pi \mid C) &\geq I_{\mu_p}(Y; \Pi \mid D, C) \\ &= \Pr[D = 0] I_{\mu_p}(Y; \Pi \mid D = 0, C) + \Pr[D = 1] I_{\mu_p}(Y; \Pi \mid D = 1, C) \\ &\geq \Pr[D = 1] I_{\mu_p}(Y; \Pi \mid D = 1, C). \end{aligned}$$

Because the distribution of Y conditioned on $D = 1$ is exactly ν, the information cost in this branch is precisely $I_\nu(Y; \Pi \mid C)$. Substituting $\Pr[D = 1] = \Theta(p \cdot t)$, we have $I_{\mu_p}(Y; \Pi \mid C) \geq \Theta(p \cdot t) \cdot I_\nu(Y; \Pi \mid C)$.

Step 4: Chain rule expansion for R. We now have a bound in terms of the unconditional information cost $I_\nu(Y; \Pi \mid C)$. However, our base lower bound from Lemma 21 bounds the *conditional* information cost $I_\nu(Y; \Pi \mid R, C)$. To bridge this gap, we observe the relationship between Y and the public randomness R under ν. If $Y = e_i$, the chosen index must be $R = i$. If $Y = 0^t$, R is uniformly distributed in $[t]$. Therefore, R can be viewed as a randomized function of Y. Consequently, $R \to Y \to \Pi$ forms another Markov chain, meaning R provides no additional information about the transcript Π once Y and C are known. Thus, $I_\nu(R; \Pi \mid Y, C) = 0$. Applying the chain rule again:

$$\begin{aligned} I_\nu(Y; \Pi \mid C) &= I_\nu(Y, R; \Pi \mid C) - I_\nu(R; \Pi \mid Y, C) = I_\nu(Y, R; \Pi \mid C) \\ &= I_\nu(R; \Pi \mid C) + I_\nu(Y; \Pi \mid R, C) \geq I_\nu(Y; \Pi \mid R, C). \end{aligned}$$

Step 5: Putting it all together. We concatenate the inequalities derived in Step 3 and Step 4, and substitute the conditional lower bound from Lemma 21:

$$\begin{aligned} I_{\mu_p}(Y; \Pi \mid C) &\geq \Theta(p \cdot t) \cdot I_\nu(Y; \Pi \mid R, C) \geq \Theta(p \cdot t) \cdot \Omega\left(\min\left(1, \frac{1}{t} \log \frac{1}{\delta}\right)\right) \\ &= \Omega\left(p \cdot \min\left(t, \log \frac{1}{\delta}\right)\right). \end{aligned}$$

This establishes the required information cost bound for the independent product distribution. ◀

4.1.3 Direct Sum for EMostlyDISJ

We now apply an information-theoretic direct sum argument, following the framework of Braverman and Zamir [7], to lift the $\Omega(p \log(1/\delta))$ lower bound from the independent 1-bit $F_{ct,t}$ instances to the full EMostlyDISJ problem. Let m be the total set size parameter across all players. To suppress spurious collisions and simulate disjointness, we assume the abstract universe U is exceptionally large, specifically $|U| \geq m^4$. We set the independent inclusion probability for our product distribution to $p = m/(2t|U|)$. Before executing the direct sum, we must guarantee that sampling sets independently under μ_p naturally generates inputs that satisfy the NO instance promise of EMostlyDISJ: namely, that the resulting sets are κ-almost-disjoint with overwhelmingly high probability.

▶ **Proposition 23.** *Let U be a universe of size $|U| \geq m^4$. Suppose t players construct sets $S_1, \ldots, S_t$ such that each S_i is chosen uniformly and independently at random from all subsets of U of size exactly m/t. With probability at least $1 - \delta$, the resulting collection of sets is κ-almost-disjoint, where $\kappa = O(\log(1/\delta))$.*

Proof. We upper bound the probability of having too many non-disjoint coordinates, which we call an *overlap* and occurs when an element $u \in U$ is chosen by two or more players. Because each of the t players draws a set of size exactly m/t, the total number of slots draws across all players is exactly m. For κ slots out of the m total slots to be non-unique, they must form at least $\kappa/2$ colliding pairs. We construct a union bound over all possible ways this could occur:

1. There are $\binom{m}{\kappa}$ ways to select which κ specific slots will be involved in the collisions.
2. For each of these κ slots, we must specify a collision partner from the m total slots, giving at most m^κ possible partner assignments.
3. Forming these partner assignments forces at least $\kappa/2$ equality constraints between the slots. We upper bound the probability of satisfying a specific set of $\kappa/2$ equalities. The total number of valid ways to assign elements to the m slots, drawing s items without replacement for each of the t players, is exactly $\left(\frac{|U|!}{(|U|-s)!}\right)^t \geq (|U|-s)^m$. The number of assignments satisfying the $\kappa/2$ equalities is at most $|U|^{m-\kappa/2}$, because the equalities group the m slots into at most $m - \kappa/2$ connected components, and each component can take at most $|U|$ values. Thus, the probability of satisfying a specific set of $\kappa/2$ equalities is at most $\frac{|U|^{m-\kappa/2}}{(|U|-s)^m} = \frac{1}{|U|^{\kappa/2}} \left(\frac{|U|}{|U|-s}\right)^m$. Since $|U| \geq m^4$ and $s \leq m$, we have $\left(\frac{|U|}{|U|-s}\right)^m = \left(1 + \frac{s}{|U|-s}\right)^m \leq e^{sm/(|U|-s)}$. Since $|U| - s \geq m^4 - m \geq m^2$, the exponent is at most $sm/(|U|-s) \leq m^2/m^2 = 1$. Therefore, the probability of satisfying any specific set of $\kappa/2$ equality constraints is at most $e/|U|^{\kappa/2}$.

Multiplying these factors together, the probability of observing at least κ overlaps is upper bounded by $\binom{m}{\kappa} m^\kappa \frac{e}{|U|^{\kappa/2}} \leq e \left(\frac{em}{\kappa}\right)^\kappa \left(\frac{m^2}{|U|}\right)^{\kappa/2} = e \left(\frac{em^2}{\kappa\sqrt{|U|}}\right)^\kappa$. Substituting our universe size assumption $|U| \geq m^4$ implies $\sqrt{|U|} \geq m^2$. Thus, the probability is at most $e \left(\frac{e}{\kappa}\right)^\kappa$. By setting $\kappa = C \log(1/\delta)$ for a sufficiently large constant C, it follows that the probability is at most $O(\delta)$. Thus, the uniformly sampled fixed-size sets are globally κ-almost-disjoint with probability at least $1 - \delta$. ◀

With the promise condition satisfied, we establish the final communication lower bound by synthesizing $|U|$ parallel games.

▶ **Corollary 24.** *Let $\kappa = O(\log(1/\delta))$ and assume $t = \Omega\left(\log \frac{1}{\delta}\right)$ and $te^{-m/(6t)} \leq \delta$. Let Π be a one-way communication protocol of worst-case transcript length C_Π, utilizing public coins C, that solves the t-party EMostlyDISJ problem with failure probability at most $O(\delta)$, total set size parameter m, almost-disjointness parameter κ, and universe size $|U| \geq m^4$. Let $\beta := \Pr[F = 1] \leq te^{-m/(6t)}$ be the abort probability. The conditional information cost of the protocol satisfies $I_\mu(\Pi; X' \mid C) \geq \frac{\Omega\left(\frac{m}{t}\log\left(\frac{1}{\delta}\right)\right) - h_2(\beta) - \beta(C_\Pi+1)}{1-\beta}$, where h_2 is the binary entropy function.*

Proof. We employ the direct sum reduction formalized in Section 4.2 of [7]. We consider $|U|$ parallel, independent instances of the 1-bit problem $F_{ct,t}$. Let the input matrix $Y \in \{0,1\}^{t\times|U|}$ consist of columns Y^j drawn i.i.d. from the product distribution μ_p, where $p = m/(2t|U|)$.

We construct a protocol Π' that solves these parallel instances by simulating the EMostlyDISJ protocol Π. Each player i locally constructs a preliminary set $S'_i = \{j \in U : Y_i^j = 1\}$. Because $Y_i^j \sim \text{Bern}(p)$, the expected size of this set is $\mathbb{E}[|S'_i|] = p|U| = m/(2t)$.

If any player's set size exceeds double its expectation ($|S'_i| > m/t$), the reduction immediately aborts and declares a failure (Event F). Otherwise, player i pads S'_i up to a size of exactly m/t using dummy elements drawn uniformly at random without replacement from the unselected elements of the common universe $U \setminus S'_i$. Let $X' = (S_1, \ldots, S_t)$ be the resulting collection of padded sets. The players then execute $\Pi(X')$.

Step 1: Bounding failure and error. By a standard Chernoff bound, the probability that a single player's set exceeds m/t is at most $e^{-m/(6t)}$. By a union bound over all t players, the probability of the failure event F is bounded by $\beta := \Pr[F = 1] \leq te^{-m/(6t)}$. By adjusting constants appropriately in our parameter assumptions, $\beta \leq \delta$. Provided F does not occur, Lemma 4.6 of [7] guarantees that the resulting compound protocol Π' correctly solves the embedded instances with high probability. The total error probability of Π' is upper bounded by $O(\delta)$. Crucially, because the initial Bernoulli process treats all elements of U symmetrically, padding uniformly from the unselected elements guarantees that the resulting set S_i is distributed uniformly over all subsets of U of size m/t. Because the players pad independently, conditioned on $F = 0$, the collection X' matches the distribution μ of independent, uniform fixed-size subsets. Because the universe size is $|U| \geq m^4$, Proposition 23 applies directly to the padded sets X'. Thus, with probability at least $1 - \delta$, the constructed sets satisfy the κ-almost-disjointness promise.

Step 2: Direct sum over independent instances. Because the columns Y^j are drawn completely independently under μ_p, the mutual information between the full matrix Y and the transcript Π' decomposes linearly via the super-additivity of independent inputs $I(Y;\Pi') \geq \sum_{j\in U} I(Y^j;\Pi')$. Applying our 1-bit lower bound from Lemma 22 to each instance, we aggregate the information $I(Y;\Pi') \geq |U| \cdot \Omega\left(p\log\left(\frac{1}{\delta}\right)\right) = |U| \cdot \Omega\left(\frac{m}{2t|U|}\log\left(\frac{1}{\delta}\right)\right) = \Omega\left(\frac{m}{t}\log\left(\frac{1}{\delta}\right)\right)$.

Step 3: Relating to the original protocol. Finally, we must relate the unconditional information cost $I(Y;\Pi')$ of the simulated inputs back to the actual conditional information cost $I_\mu(\Pi; X') = I(\Pi'; X' \mid F = 0)$ of the base protocol operating on the padded sets.

Conditioned on $F = 0$, the symmetric random padding from the common universe U ensures the padded sets match the distribution μ of independent uniform fixed-size subsets. Moreover, conditioned on X', $F = 0$, and the public coins C, the transcript is generated only from X' and the protocol's private coins. Hence, $Y \to X' \to \Pi'$ is a Markov chain,

which implies by the data processing inequality $I(Y;\Pi' \mid F=0,C) \le I(X';\Pi' \mid F=0,C) = I_\mu(\Pi;X' \mid C)$. We expand the mutual information $I(Y;\Pi' \mid C)$ using the chain rule over the abort indicator F:

$$\begin{aligned} I(Y;\Pi' \mid C) &\le I(Y;\Pi',F \mid C) = I(Y;F \mid C) + I(Y;\Pi' \mid F,C) \\ &\le h_2(\beta) + (1-\beta)I(Y;\Pi' \mid F=0,C) + \beta I(Y;\Pi' \mid F=1,C). \end{aligned}$$

Let C_Π be the worst-case bit length of the transcript of Π. Even on an abort ($F=1$), we can pad the compound transcript to length $C_\Pi+1$. Thus, the conditional mutual information $I(Y;\Pi' \mid F=1)$ is upper bounded by $C_\Pi+1$. Substituting these bounds into our expansion, we have $I(Y;\Pi' \mid C) \le h_2(\beta) + (1-\beta)I_\mu(\Pi;X' \mid C) + \beta(C_\Pi+1)$. Rearranging the terms to isolate $I_\mu(\Pi;X')$, we have $I_\mu(\Pi;X' \mid C) \ge \frac{I(Y;\Pi'|C) - h_2(\beta) - \beta(C_\Pi+1)}{1-\beta}$. Substituting the main term $I(Y;\Pi' \mid C) \ge \Omega\left(\frac{m}{t}\log\left(\frac{1}{\delta}\right)\right)$ from Step 2, we obtain the lower bound $I_\mu(\Pi;X' \mid C) \ge \frac{\Omega\left(\frac{m}{t}\log\left(\frac{1}{\delta}\right)\right) - h_2(\beta) - \beta(C_\Pi+1)}{1-\beta}$. This concludes the direct sum. ◀

4.2 F_2 Estimation Lower Bound via Multi-Scale Reduction

We now employ the multi-scale reduction framework of [7] (Section 5) using EMostlyDISJ as the building block.

4.2.1 Reduction from EMostlyDISJ to F_2

We first generalize the reduction in Section 5.1 of [7] to EMostlyDISJ and allow for flexible parameters t. Note that this section describes the construction for a *single level* (i.e., a single instance of the game played with exactly t players); in the next section, we will embed multiple such levels simultaneously. Fix n,ε. Let $2 \le t \le \varepsilon\sqrt{n}/2$. Define the super-element size $d := \lfloor \frac{\varepsilon^2 n}{4t^2} \rfloor \ge 1$. We reduce from a single instance of EMostlyDISJ$_t$. The abstract universe of items for this game is instantiated as U^d (where each abstract item is a super-element, defined as a d-tuple of distinct elements from a base stream universe U). The total set size parameter m from the previous section (which bounds the total number of abstract items distributed across all players' sets) is set to $m = \lfloor n/(4d) \rfloor = \Theta(t^2/\varepsilon^2)$. We set the almost-disjointness parameter to $\kappa = O(\log(1/\delta))$.

Given an instance $X = (S_1,\dots,S_t,x) \in (U^d)^t \times U^d$, we construct a stream $s(X)$ of base items from U. This implicitly defines a frequency vector $v \in \mathbb{N}^{|U|}$, where each coordinate v_u records the exact number of times base item $u \in U$ appears in the stream. First, we concatenate the elements of the players' sets S_i: each super-element is "unzipped" and its d distinct constituent base elements from U are written to the stream. Since there are at most m super-elements across all players, this contributes $md \le n/4$ base items to the stream. Finally, we append the referee's target super-element x by writing its d base elements repeatedly $k = \lceil t/\varepsilon \rceil$ times. Because $k = \lceil t/\varepsilon \rceil$ and $d \le \frac{\varepsilon^2 n}{4t^2}$, we have $dk \approx \frac{\varepsilon n}{4t} \le n/4$. The total stream length is upper bounded by $md + dk \le n/4 + n/4 \le n$.

▶ **Definition 25** (ζ-Base-Separation). *A stream built from an instance X is ζ-base-separated if the number of non-unique base-element memberships arising from coordinates of distinct abstract items is at most ζ.*

▶ **Lemma 26** (Adapted from Lemma 5.2 of [7]). *If $t \ge C_1 \log\frac{1}{\delta}$ and $\kappa \le C_2 \log\frac{1}{\delta}$ for sufficiently large constants $C_1 > C_2$, and the stream is ζ-base separated for $\zeta \le C_2 \log\frac{1}{\delta}$, then a $(1 \pm \Theta(\varepsilon))$-approximation to $F_2(s(X))$ solves any valid instance X of EMostlyDISJ$_t$ where the total number of base element collisions is at most κ.*

Proof. We analyze the gap in the second frequency moment $F_2(v) = \sum_{u \in U} v_u^2$ between YES and NO instances. For simplicity, we assume $c = 1/2$ and $\gamma \leq \kappa$. Let F_{bg} denote the F_2 contribution from the background base items (all items not belonging to the referee's target x or the heavy element j_0). By assumption, the total number of base element collisions is at most κ. Because the m super-elements contribute $md \leq n/4$ base items, then we have $F_{bg} \leq d(\kappa+1)^2 + n/4 \leq n/4 + \mathcal{O}(d\kappa^2)$. We now distinguish the YES instance from two logically distinct NO instances.

In the first NO case, the input sets are κ-almost disjoint, and the referee's target x is repeated $k = \lceil t/\varepsilon \rceil$ times. The target x appears at most κ times among the players. Including additional collisions, the d coordinates of the target reach a frequency of at most $k + 2\kappa$. Hence, we have $F_2(\mathrm{NO}_1) \leq F_{bg} + d(k+2\kappa)^2 \leq n/4 + dk^2 + O(dk\kappa + d\kappa^2)$.

In the second NO case, there is a heavy element j_0 appearing $t/2$ times, but the referee's target is a different element x appearing k times. Because x and j_0 are distinct super-elements, their base coordinates are completely disjoint except for the additional collisions, which occur at most κ times. Thus, the d coordinates of j_0 have frequency at most $t/2 + \kappa$, and the d coordinates of x have frequency at most $k + \kappa$. Hence, we have $F_2(\mathrm{NO}_2) \leq F_{bg} + d(t/2+\kappa)^2 + d(k+\kappa)^2 \leq n/4 + d\left(\frac{t^2}{4} + k^2\right) + O(dt\kappa + dk\kappa + d\kappa^2)$.

In a YES instance, the heavy element j_0 is exactly the referee's target x. The d target coordinates appear at least $t/2$ times from the players and k times from the referee, reaching a frequency of $t/2+k$. Then we have $F_2(\mathrm{YES}) \geq F'_{bg} + d\cdot(t/2+k)^2 \geq n/4 - d(t/2) + d\cdot(t/2+k)^2$. Here, $F'_{bg} \geq n/4 - d(t/2)$ accounts for the target items being removed from the background calculation, assuming the total size of the players' sets is exactly m.

Observe that the F_2 moment for the NO instance is maximized in Case 2, and so the additive gap between the YES case and the NO case is at least:

$$\begin{aligned}
\Delta F_2 &= F_2(\mathrm{YES}) - F_2(\mathrm{NO}_2) \\
&\geq d \cdot [(t/2+k)^2 - (t/2+\kappa)^2 - (k+\kappa)^2] - O(dt\kappa + dk\kappa + d\kappa^2) - dt/2 \\
&= d \cdot [(t/2)^2 + tk + k^2 - (t/2)^2 - t\kappa - \kappa^2 - k^2 - 2k\kappa - \kappa^2] - O(dt\kappa + dk\kappa + d\kappa^2) - dt/2 \\
&= d \cdot [tk - O(t\kappa) - O(k\kappa) - O(\kappa^2) - O(t)].
\end{aligned}$$

Substituting $k = \lceil t/\varepsilon \rceil$ and $d = \lfloor \varepsilon^2 n/(4t^2) \rfloor$, we have $\Delta F_2 \geq \frac{\varepsilon^2 n}{4t^2} \cdot \left[\frac{t^2}{\varepsilon} - O\left(\frac{\kappa t}{\varepsilon}\right)\right]$. The dominant term is $\frac{\varepsilon^2 n}{4t^2} \cdot \frac{t^2}{\varepsilon} = \frac{\varepsilon n}{4}$. The subtracted negative terms are negligible, since $t^2/\varepsilon \gg \kappa t/\varepsilon$ for $t \geq C_1 \log \frac{1}{\delta}$ and $\kappa \leq C_2 \log \frac{1}{\delta}$ for sufficiently large constants $C_1 > C_2$. We also need $t^2/\varepsilon \gg t^2$, which holds for $\varepsilon < 1$. If $t \geq C \log(1/\delta)$, the additive gap remains $\Delta F_2 = \Omega(\varepsilon n)$.

To establish that this additive gap translates to a strict $(1+\Omega(\varepsilon))$-multiplicative factor, we bound the maximum value of $F_2(\mathrm{NO}_2)$. By our parameter choices, the referee's isolated squared contribution is $dk^2 \approx \left(\frac{\varepsilon^2 n}{4t^2}\right)\frac{t^2}{\varepsilon^2} = \Theta(n)$. Since $F_{bg} \leq n/4 + O(\kappa^2)$, it follows that $F_2(\mathrm{NO}_2) \leq n/4 + \Theta(n) + o(n) = \Theta(n)$. Since the baseline is $F_2(\mathrm{NO}_2) = \Theta(n)$ and the additive gap is $\Delta F_2 = \Omega(\varepsilon n)$, the multiplicative ratio between the instances is $\frac{F_2(\mathrm{YES})}{F_2(\mathrm{NO}_2)} \geq \frac{F_2(\mathrm{NO}_2)+\Omega(\varepsilon n)}{F_2(\mathrm{NO}_2)} = 1 + \Omega\left(\frac{\varepsilon n}{n}\right) = 1 + \Omega(\varepsilon)$. Because the YES and NO cases differ by a $(1+\Omega(\varepsilon))$-multiplicative factor, any streaming algorithm providing a $(1 \pm c\varepsilon)$-relative approximation to F_2 (for a sufficiently small constant c) will successfully distinguish between the two cases, solving the single-level EMostlyDISJ$_t(X)$ instance. ◀

A streaming algorithm $\mathcal{A}$ for F_2 naturally induces a communication protocol Π for EMostlyDISJ$_t$, where player i simulates $\mathcal{A}$ on their part of the stream and passes the memory state to player $i+1$. The total communication is $t \cdot \mathrm{Space}(\mathcal{A})$.

4.2.2 The Multi-Scale Construction and Direct Sum

We now employ the multi-scale reduction framework of [7] (Section 5.2). We define levels l, where $t_l = 2^l$, and conceptually embed an instance of EMostlyDISJ$_{t_l}$ at each level.

We define a range of "good" levels. A level l is good if t_l satisfies the necessary conditions for both the F_2 reduction and the communication lower bound:

1. $t_l \leq \varepsilon\sqrt{n}/2$ (Ensures the super-element size $d_l \geq 1$).
2. $t_l \geq C\log(1/\delta)$ (Ensures the F_2 gap is distinguishable, Lemma 26).
3. $\delta \geq t_l e^{-m_l/(6t_l)}$ (Condition for Corollary 24). Since $m_l/t_l = \Theta(t_l/\varepsilon^2)$, this requires $t_l/\varepsilon^2 \geq \Omega(\log(t_l/\delta))$.

The number of good levels L is approximately $\log_2(\varepsilon\sqrt{n}) - \log_2(C'\max(1,\varepsilon^2)\log(1/\delta))$. Assuming standard parameter regimes where ε is small, this yields $L = \Omega(\log(\frac{\varepsilon\sqrt{n}}{\log(1/\delta)}))$. A naïve concatenation of these instances would result in a stream of length $\omega(n\log n)$, which would artificially inflate the space bounds. To embed these instances without increasing the total stream length beyond n, the construction interleaves them implicitly within a single continuous stream. The information cost is analyzed over a stream drawn from a distribution $\mathcal{E}$, where n base elements are chosen uniformly and independently from U.

To upper bound element repetitions across instances, we set the base universe size to $|U| = n^3$. Let $\mathcal{E}_{coll}$ be the event that there are fewer than $\kappa = C_1\log\frac{1}{\delta}$ collisions in the entire stream of length n. Note that since the expected number of collisions is $\mu \leq \binom{n}{2}/n^3 < \frac{1}{2n}$, then by standard tail bounds, the probability of κ or more collisions is at most $O(\delta)$. Hence, for the remainder of the section, we condition on the event $\mathcal{E}_{coll}$. Now, because the stream consists of independent random variables, it can be analyzed at multiple scales simultaneously. For any level l, partitioning the stream into contiguous blocks of size n/t_l to form d_l-tuples naturally generates a valid NO instance for the t_l-player EMostlyDISJ game. However, before bounding the information measure, we must reconcile a subtle distributional difference between the multiscale stream and the communication game of Corollary 24. The communication inputs X' are fixed-size sets drawn *without replacement* from the universe U^{d_l}, whereas the continuous multiscale stream generates d_l-tuples *with replacement*.

▶ **Lemma 27** (Distributional Coupling). *Let $\mathcal{D}_{stream}$ be the distribution of a single active stream block at level l, consisting of m_l abstract d_l-tuples drawn i.i.d. with replacement from U^{d_l}. Let $\mathcal{D}_{game}$ be the exact communication distribution from Corollary 24, which generates sets without replacement. For any streaming algorithm's memory state M processing the stream $X \sim \mathcal{D}_{stream}$, the mutual information satisfies $I(X;M) \geq \left(1-\frac{1}{n}\right)\cdot I(X_{game};M)$, where $X_{game} \sim \mathcal{D}_{game}$.*

Proof. Let E be the event that the sequence of m_l items drawn from U^{d_l} contains no duplicate tuples within each player. Conditioned on E, a sequence drawn i.i.d. with replacement is identical to a uniform sequence drawn without replacement. Thus, $(X \mid E) \sim \mathcal{D}_{\text{game}}$. Because we set the base universe size to $|U| \geq n^3$ and $d_l \geq 1$, the abstract universe size is $|U^{d_l}| = |U|^{d_l} \geq n^3$. The total number of abstract items drawn across all t_l players in the instance is $m_l \leq n$. Hence, the probability of drawing a duplicate is at most $\Pr[\neg E] \leq \frac{m_l^2}{2|U^{d_l}|} \leq \frac{n^2}{2n^3} \leq \frac{1}{2n} \leq \frac{1}{n}$. To lower bound the information cost, we use the chain rule for mutual information. Let I_E be the binary indicator of event E. Expanding $I(X, I_E; M)$ in two ways, we have $I(X;M) + I(I_E;M \mid X) = I(I_E;M) + I(X;M \mid I_E)$. Since I_E is a deterministic function of X, we have $H(I_E \mid X) = 0$ and hence $I(I_E;M \mid X) = 0$. Since mutual information is non-negative, $I(I_E;M) \geq 0$. Thus $I(X;M) \geq I(X;M \mid I_E)$. Expanding the conditional mutual information over the states of I_E, $I(X;M \mid I_E) =$

$\Pr[E] \cdot I(X; M \mid E) + \Pr[\neg E] \cdot I(X; M \mid \neg E) \geq \Pr[E] \cdot I(X; M \mid E)$. Conditioned on E, the input is distributed as $\mathcal{D}_{\text{game}}$ and the memory state is produced by the same algorithm, so the joint law of (X, M) given E equals that of (X_{game}, M); hence $I(X; M \mid E) = I(X_{\text{game}}; M)$. Substituting $\Pr[E] \geq 1 - 1/n$ gives the desired claim. ◀

We use the information measures defined by [7]. Let X be the random stream and M_j the memory state of the algorithm after processing index j.

▶ **Definition 28** (Definition 5.6 of [7]). *For level l, with $t_l = 2^l$, define the information measure $I_l := \sum_{j=1}^{n} I\left(X_{(j-\frac{n}{2^l}, j-\frac{n}{2^{l+1}})}; M_j \mid M_{j-\frac{n}{2^l}}\right)$.*

This measure sums over all time steps j to capture the total information the algorithm must retain about the past stream segments relevant to scale l.

▶ **Lemma 29** (Adapted from Lemma 5.7 of [7]). *Let S be the worst-case space (in bits) of the streaming algorithm. For each good level l, the information measure I_l satisfies $I_l \geq \Omega\left(\frac{n}{\varepsilon^2} \log \frac{1}{\delta}\right) - O(n\beta_l S) - O\left(\frac{n}{t_l} h_2(\beta_l)\right)$.*

Proof. Following the structure of Lemma 5.7 in [7], we lower bound I_l by tracking the number of independent instances of EMostlyDISJ_{t_l} that can be embedded within the stream. For a fixed level l, simulating a single instance of the game requires t_l players sequentially passing the memory state of the streaming algorithm, corresponding to t_l stream blocks of length n/t_l. Because the streaming algorithm uses at most S bits of space, the induced communication protocol has a worst-case transcript length $C_{\Pi_l} \leq t_l S$. Evaluating this instance requires examining the sequence of memory states M_j spaced at intervals of exactly n/t_l. However, this sequence of memory states need not begin at index 1. Because the stream elements are drawn independently, shifting the starting index of these intervals yields disjoint set of indices j defining I_l, and thus independent communication games. Since this starting offset can be shifted exactly n/t_l times before the blocks begin to overlap, [7] identified $\Omega(n/t_l)$ disjoint sequences of indices $j_1, \ldots, j_{t_l}$ that serve as independent parallel transcripts for the EMostlyDISJ_{t_l} instances at that level. By Corollary 24 (and noting $1/(1-\beta_l) \leq O(1)$ since $\beta_l \leq \delta \leq 1/2$), the conditional information cost to solve one such instance is $\Omega(\frac{m_l}{t_l} \log(1/\delta)) - O(\beta_l C_{\Pi_l}) - O(h_2(\beta_l))$. Substituting $m_l = \Theta(t_l^2/\varepsilon^2)$ from our parameter choice and $C_{\Pi_l} \leq t_l S$, the cost per instance becomes $\Omega(\frac{t_l}{\varepsilon^2} \log(1/\delta)) - O(\beta_l t_l S) - O(h_2(\beta_l))$.

To lower bound the information cost of a single embedded instance, we apply the conditioning argument required by Lemma 27. Let $X \sim \mathcal{D}_{\text{stream}}$ be the stream instance. The coupling guarantees $I(X; \Pi) \geq \left(1 - \frac{1}{n}\right) \cdot I(X_{\text{game}}; \Pi)$. Because the streaming algorithm guarantees a failure probability of at most δ on *any* valid input sequence, it fails on the communication game on the conditional distribution X_{game} with probability less than δ. Thus, Corollary 24 applies directly to $I(X_{\text{game}}; \Pi)$, so that $I(X_{\text{game}}; \Pi) \geq \Omega\left(\frac{m_l}{t_l} \log \frac{1}{\delta}\right) - O(\beta_l C_{\Pi_l}) - O(h_2(\beta_l))$. Substituting $m_l = \Theta(t_l^2/\varepsilon^2)$ and $C_{\Pi_l} \leq t_l S$, the expected information cost per instance under $\mathcal{D}_{\text{stream}}$ is lower bounded by $\left(1 - \frac{1}{n}\right)\left[\Omega\left(\frac{t_l}{\varepsilon^2} \log \frac{1}{\delta}\right) - O(\beta_l t_l S) - O(h_2(\beta_l))\right]$.

Now, using Lemma 3.2 of [7] (a chain rule for streaming mutual information), the total information cost I_l is lower bounded by the sum of the costs of these independent interleaved instances. Therefore:

$$
\begin{aligned}
I_l &\geq (\text{Number of independent instances}) \times (\text{Information cost per instance}) \\
&\geq \Omega\left(\frac{n}{t_l}\right) \times \left[\Omega\left(\frac{t_l}{\varepsilon^2} \log \frac{1}{\delta}\right) - O(\beta_l t_l S) - O(h_2(\beta_l))\right] \\
&= \Omega\left(\frac{n}{\varepsilon^2} \log \frac{1}{\delta}\right) - O(n\beta_l S) - O\left(\frac{n}{t_l} h_2(\beta_l)\right).
\end{aligned}
$$

Observe that the dependence on t_l cancels out in the main term. This gives a uniform lower bound on the required information cost across all good levels, independent of the number of players involved at scale l. ◀

The final step is the innovative direct sum argument over dependent instances developed by [7]. They define the average information stored by the algorithm:

▶ **Definition 30** (Definition 5.9 of [7]). $\bar{I} := \frac{1}{n}\sum_{j=1}^{n} I(X_{<j}; M_j)$.

They prove that the information measures I_ℓ from different scales essentially add up, even though the underlying problems are dependent.

▶ **Lemma 31** (Lemma 5.10 of [7]). $\bar{I} \geq \frac{1}{n}\sum_{\ell \in Good} I_\ell$.

Proof. This follows from Lemma 5.8 of [7], which uses the chain rule to show that for any j, $I(X_{<j}; M_j)$ is lower bounded by the sum over ℓ of the information M_j has about the specific past segment relevant to level ℓ, conditioned on the memory state corresponding to the beginning of that segment. This relies on the observation that these past segments are disjoint for different ℓ. Summing over j and reordering yields the result. ◀

Proof of Theorem 1. Let S be the space required by the algorithm. This space is at least the maximum information stored, which is at least the average information $\bar{I}$ (Observation 3.1 in [7]), so $S \geq \bar{I}$. By combining Lemma 29 and Lemma 31:

$$S \geq \bar{I} \geq \frac{1}{n}\sum_{\ell \in \text{Good}} I_\ell \geq \frac{1}{n}\sum_{\ell \in \text{Good}} \left[\Omega\left(\frac{n}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right) - O(n\beta_\ell S) - O\left(\frac{n}{t_\ell}h_2(\beta_\ell)\right)\right]$$

$$= (\text{Number of Good Levels}) \cdot \Omega\left(\frac{1}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right) - S\sum_{\ell \in \text{Good}} O(\beta_\ell) - \sum_{\ell \in \text{Good}} O\left(\frac{h_2(\beta_\ell)}{t_\ell}\right).$$

The number of good levels is $L = \Omega(\log(\frac{\varepsilon\sqrt{n}}{\log(1/\delta)}))$. Observe that the abort probabilities $\beta_\ell \leq t_\ell e^{-m_\ell/(6t_\ell)}$ decay exponentially across the dyadic levels since $m_\ell/t_\ell = \Theta(t_\ell/\varepsilon^2)$. Thus, by choosing constants appropriately, we ensure that the aggregated defect sum is bounded by a fraction: $\sum_{\ell \in \text{Good}} O(\beta_\ell) \leq 1/2$. The entropy penalty sum is similarly upper bounded by a negligible constant $O(1)$. Substituting these bounds into our inequality yields $S \geq \Omega\left(\log\left(\frac{\varepsilon\sqrt{n}}{\log\frac{1}{\delta}}\right)\frac{1}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right) - \frac{1}{2}S - O(1)$. Thus, we conclude $S \geq \Omega\left(\log\left(\frac{\varepsilon\sqrt{n}}{\log\frac{1}{\delta}}\right)\frac{1}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\right)$. ◀

5 Improved Algorithms

5.1 Bounded Frequencies

We present an ℓ_2-estimation algorithm for insertion-only streams where the frequencies are bounded by B. The goal is to achieve a space complexity where the dependence on B is only logarithmic, and the dependence on n is also logarithmic, independent of $1/\varepsilon^2$.

▶ **Theorem 32.** *Let $x \in \mathbb{Z}^n_{\geq 0}$ be the frequency vector of an insertion-only stream of length m. Suppose that $0 \leq x_i \leq B$ for all $i \in [n]$. Then there exists a streaming algorithm that, with probability at least $1-\delta$, computes a $(1\pm\varepsilon)$-approximation to $\|x\|_2^2$ using space (in bits) $O\left(\frac{1}{\varepsilon^2}\log^2\frac{B}{\varepsilon}\log\frac{1}{\delta}\left(\log B + \log\frac{1}{\varepsilon}\right)\right) + O\left(\log\frac{n}{\delta}\right)$.*

Proof. The strategy is to use subsampling to reduce the effective dimension of the problem to a size that depends polynomially on B and $\frac{1}{\varepsilon}$, but only logarithmically on n. Then, we apply an efficient ℓ_2 estimator on this reduced stream. We use pairwise independent hash

functions $h : [n] \to [2^L]$ (where $L = O(\log n)$). We define levels of sampling based on the number of trailing zeros in the hash value $h(i)$. Let $z(i)$ be the number of trailing zeros of $h(i)$. We aim to find a sampling level ℓ such that the expected number of distinct items i with $z(i) \geq \ell$ is approximately K, where $K = \Theta\left(\frac{B^2}{\varepsilon^2}\right)$, independent of δ.

We first utilize a continuous F_0 tracking algorithm [23] to estimate the total number of distinct elements F_0. By setting the failure probability of the F_0 estimator to $\delta/\log n$, we may assume that it yields a constant-factor approximation to F_0 at all times throughout the stream. Let $F_{0,t}$ be the number of non-zero entries at time t of the stream, and let $\hat{F}_{0,t}$ be the corresponding estimator. We first describe a two-pass version of our algorithm, and then explain how to extend it to a single pass.

A two-pass algorithm. On the first pass, we compute the final estimate $\hat{F}_0$. We choose the sampling level ℓ such that $2^\ell \approx \hat{F}_0/K$. This ensures that the expected number of items surviving the sampling is within a constant factor of K. Let S be the set of surviving coordinates: $S = \{i \in [n] \mid z(i) \geq \ell\}$. The expected size of S is roughly $F_0/2^\ell \approx K$.

We define the natural inverse-probability estimator based on the sampled set S, so that $Y = 2^\ell \sum_{i \in S} x_i^2 = \frac{1}{p} \sum_{i \in S} x_i^2$, where $p = 2^{-\ell}$. Since Y is a standard sampling estimator and elements are sampled pairwise independently, its expectation is $\mathbb{E}[Y] = \|x\|_2^2$, and its variance is upper bounded by $\mathrm{Var}(Y) \leq \frac{1-p}{p} \sum_{i=1}^n x_i^4 \leq \frac{1}{p}\|x\|_4^4$. Since the frequencies are upper bounded by B ($x_i \leq B$), we have $\|x\|_4^4 \leq B^2\|x\|_2^2$, which implies $\mathrm{Var}(Y) \leq \frac{B^2}{p}\|x\|_2^2$.

To achieve our $(1 \pm \varepsilon)$ guarantee, Chebyshev's inequality implies that it suffices to have $\mathrm{Var}(Y) \leq \varepsilon^2\|x\|_2^4$. This holds if $p \geq \Omega\left(\frac{B^2}{\varepsilon^2\|x\|_2^2}\right)$. Since all non-zero entries are at least 1, $\|x\|_2^2 \geq F_0$. Therefore, it suffices to set $p \geq \Omega\left(\frac{B^2}{\varepsilon^2 F_0}\right)$. Our choice of ℓ, which ensures $p \approx K/\hat{F}_0$, guarantees this condition when the constants are set appropriately. Thus, a single sampling instance gives a $(1 \pm \varepsilon)$-approximation with constant probability, e.g., $7/8$. Finally, to achieve the required $1 - \delta$ success probability, confidence bound, we utilize the standard median-of-means technique, taking the median of $T = \Theta(\log(1/\delta))$ independent instances.

Implementation and space complexity. We do not explicitly store the sampled set S_r for each independent instance r. Instead, for each of the T independent instances, we observe that the stream restricted to the coordinates in S_r is itself a valid sparse stream. To avoid paying a logarithmic dependence on n for the sketch hash seeds, we apply dimensionality reduction. Let $M = \Theta(K^2)$. We draw a pairwise independent hash function $g_r : [n] \to [M]$. Because the expected support size of the restricted stream is K, hashing into $M = \Theta(K^2)$ buckets yields zero collisions with high constant probability. Let $y^{(r)}$ be this compressed vector. When a stream update (i, Δ) arrives, we evaluate $h_r(i)$ to check if $z_r(i) \geq \ell$. If so, we pass the update to a standard AMS sketch [2] tracking $x|_{S_r}$. Because $K = \Theta(B^2/\varepsilon^2)$ is independent of δ, the expected ℓ_1 norm of each restricted substream $x|_S$ is at most $BK = \Theta(B^3/\varepsilon^2)$. By standard concentration bounds, the maximum ℓ_1 norm is $O(BK)$ with high probability. Thus, the bit complexity for a single counter is at most $\log(BK) = O(\log B + \log(1/\varepsilon))$. Each of the $T = \Theta(\log(1/\delta))$ independent instances maintains an AMS sketch with $O(1/\varepsilon^2)$ counters, operating on the reduced domain $[M]$. Thus, each of the $O(1/\varepsilon^2)$ hash functions uses $O(\log M) = O(\log K) = O(\log B + \log(1/\varepsilon))$ bits to store. Hence, the overall space required for the AMS sketches is $O\left(\frac{1}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\left(\log B + \log\frac{1}{\varepsilon}\right)\right)$. We also need a constant-factor approximation to determine our sampling probability, through a continuous monitoring F_0 algorithm that uses an additive $O\left(\log\frac{n}{\delta}\right)$ bits, c.f., Theorem 17.

Extension to a single pass. To extend this to a single pass, one could run $O(\log n)$ instances of the above two-pass algorithm in parallel, using every possible power-of-two sampling rate p. At the end of the stream, we would query $\hat{F}_0$ and output the estimate from the instance that used the correct value of p. However, this would increase the overall space by an undesired $\log n$ factor.

To avoid this overhead, we utilize our continuous F_0 tracker. Let $t_0 < t_1 < \dots$ be the timestamps at which the F_0 estimate crosses consecutive powers of two, meaning $\hat{F}_{0,t_i} \geq 2^i$. Let $N = \lceil \log(B^2/\varepsilon^2) \rceil$. At each milestone t_i, we initialize a new group of $N+1$ instances of our sketch, which consists of the $T = \Theta(\log(1/\delta))$ independent instances necessary for the median-of-means aggregation. These $N+1$ instances use sampling probabilities p calibrated for future F_0 scales of $2^0\hat{F}_{0,t_i}, 2^1\hat{F}_{0,t_i}, \dots, 2^N\hat{F}_{0,t_i}$. Crucially, we do not store every sketch indefinitely. We maintain only the sketch groups generated at the $N+1$ most recent milestones (from t_i down to t_{i-N}). Any older sketches are permanently discarded. Consequently, we maintain at most $O(N^2) = O(\log^2(B/\varepsilon))$ active sketches at any point in time.

At the end of the stream, we examine the earliest remaining group of non-discarded sketches. Among the N sketches in this group, we select the one that used the correct sampling rate p corresponding to our final F_0 estimate. This selected sketch guarantees a $(1 \pm \varepsilon)$-approximation to F_2, but only for the *suffix* of the stream processed after the sketch was initialized. Let x_2 be the frequency vector corresponding to this suffix, and let x_1 be the discarded prefix, so that $x = x_1 + x_2$. Because we retained sketches stretching back $N = \lceil \log(B^2/\varepsilon^2) \rceil$ epochs, the support size of the unsketched prefix x_1 satisfies $\|x_1\|_0 \leq 2^{-N}\|x\|_0 \leq \frac{\varepsilon^2}{B^2}\|x\|_0$. Exploiting the bounded-frequency assumption ($x_i \leq B$) and the fact that all non-zero entries are at least 1 (so $\|x\|_0 \leq \|x\|_2^2$), we can bound the ℓ_2 norm of the prefix $\|x_1\|_2^2 \leq B^2\|x_1\|_0 \leq B^2\left(\frac{\varepsilon^2}{B^2}\|x\|_0\right) = \varepsilon^2\|x\|_0 \leq \varepsilon^2\|x\|_2^2$. Taking the square root yields $\|x_1\|_2 \leq \varepsilon\|x\|_2$. By the triangle inequality, the difference between the full norm $\|x\|_2$ and the estimated suffix norm $\|x_2\|_2$ is at most $\|x_1\|_2 \leq \varepsilon\|x\|_2$. Adjusting ε by a constant factor ensures a strict $(1\pm\varepsilon)$-approximation for the final estimate. The total space complexity is dominated by the F_0 tracker plus the $O(\log^2(B/\varepsilon))$ active epoch states, where each state maintains $T = \Theta(\log(1/\delta))$ independent AMS instances. Multiplying the space complexity for each instance by these factors gives the final desired bound. ◀

5.2 Sparse Streams

We present an ℓ_2-estimation algorithm for insertion-only streams where the frequency vector x is k-sparse, meaning it has at most k non-zero entries (i.e., $\|x\|_0 \leq k$). This algorithm achieves a space complexity where k and the stream length m appear only inside logarithmic factors. The core idea is to first use hashing to reduce the dimensionality of x. We then apply the standard AMS sketch to the resulting compressed vector. To aggressively optimize the space required per counter, we do not maintain the exact sketch counters. Instead, we conceptually split each counter into positive and negative increments and maintain them approximately using Morris counters.

▶ **Theorem 33.** *Let $x \in \mathbb{Z}^n_{\geq 0}$ be the frequency vector of an insertion-only stream of length m. Suppose that $\|x\|_0 \leq k$. Then there exists a streaming algorithm that, with probability at least $1-\delta$, computes a $(1 \pm \varepsilon)$-approximation to $\|x\|_2^2$ using space (in bits) $O\left(\frac{1}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\left(\log\frac{k}{\varepsilon} + \log\log m\right) + \log n \cdot \log\frac{1}{\delta}\right)$.*

Proof. The strategy involves a two-stage process: dimensionality reduction tailored specifically for sparse vectors, followed by an AMS sketch where the underlying counters are maintained approximately using Morris counters [27].

Dimensionality reduction via hashing. We first reduce the dimension by hashing the coordinates of x into a much smaller set of random buckets. We draw $T = \Theta(\log(1/\delta))$ independent pairwise independent hash functions $h_r : [n] \to [M]$, where $M = \lceil Ck/\varepsilon^2 \rceil$ for a sufficiently large constant C. For each independent instance $r \in [T]$, we define the reduced vector $y^{(r)} \in \mathbb{R}^M$, where $y^{(r)}_j = \sum_{i:h_r(i)=j} x_i$. Since x is k-sparse, the expected ℓ_2^2 error introduced by collisions in a single instance is at most $\|x\|_1^2/M \le k\|x\|_2^2/M$. By Markov's inequality, a single hash preserves $\|y^{(r)}\|_2^2 = (1 \pm \varepsilon/4)\|x\|_2^2$ with constant probability. Taking the median across the T independent instances amplifies this probability to the desired $1 - \delta$.

AMS Sketch with Morris Counters. For each of the T instances, we apply an AMS sketch to the lower-dimensional vector $y^{(r)}$ using $R = \Theta(1/\varepsilon^2)$ estimators. We conceptually decompose each exact AMS counter into its positive and negative accumulating parts, i.e., $A_{r,w} = A^+_{r,w} - A^-_{r,w}$. Notice that the total accumulated mass across both parts is precisely the ℓ_1 norm of x: $A^+_{r,w} + A^-_{r,w} = \|x\|_1 = m$. To avoid the space overhead of storing these counters exactly, we maintain them approximately. We instantiate two unbiased Generalized Morris counters for each index (r, w), one for $A^+_{r,w}$ and one for $A^-_{r,w}$, and each with variance $\gamma = \Theta(\varepsilon^2/k^2)$. Let $\hat{A}^+_{r,w}$ and $\hat{A}^-_{r,w}$ be their respective estimates. Our approximate counter is then $\hat{A}_{r,w} = \hat{A}^+_{r,w} - \hat{A}^-_{r,w}$, which remains unbiased, i.e., $\mathbb{E}[\hat{A}_{r,w}] = A_{r,w}$. Thus, the variance of the approximate counter is at most $\mathrm{Var}(\hat{A}_{r,w}) = \mathrm{Var}(\hat{A}^+_{r,w}) + \mathrm{Var}(\hat{A}^-_{r,w}) \le \gamma(A^+_{r,w})^2 + \gamma(A^-_{r,w})^2 \le \gamma m^2$.

Error analysis. Let $\hat{Y}'_r$ be the AMS estimator computed using these approximate counters for a single instance r, i.e., averaging $R = \Theta(1/\varepsilon^2)$ squared approximate counters $\hat{A}^2_{r,w}$ across $w \in [R]$. Because the Morris counters are unbiased, the expected value of a squared approximate counter $\hat{A}^2_{r,w}$ includes the true squared signal plus the variance, i.e., $\mathbb{E}[\hat{A}^2_{r,w}] = A^2_{r,w} + \mathrm{Var}(\hat{A}_{r,w})$. Thus, the expected bias introduced to the final estimator by the approximate counters is at most $\mathrm{Var}(\hat{A}_{r,w}) \le \gamma m^2 \le O(\varepsilon^2 m^2/k^2)$. Because x is k-sparse and $\|x\|_1 = m$, then $\|x\|_2^2 \ge m^2/k$, so that the bias is at most $O(\varepsilon^2/k) \cdot \|x\|_2^2$.

Similarly, we bound the variance of the overall estimator. Let $E = \hat{A}_{r,w} - A_{r,w}$ be the error of the Morris counter. Expanding the squared counter, we have $\hat{A}^2_{r,w} = (A_{r,w} + E)^2 = A^2_{r,w} + 2A_{r,w}E + E^2$. Since $A^2_{r,w} = O(m^2)$ and $\mathrm{Var}(E) \le \gamma m^2$, then the variance is at most $O(\gamma m^4)$. Averaging over the $R = \Theta(1/\varepsilon^2)$ and applying the above argument, it follows that the variance is at most $O(\varepsilon^4\|x\|_2^4) \ll \varepsilon^2\|x\|_2^4$. Thus Chebyshev's inequality, each instance outputs a $(1 \pm \varepsilon/2)$ approximation to $\|y^{(r)}\|_2^2$ with constant probability. By taking the median of $O(\log 1/\delta)$ instances, we can boost the success probability to $1 - \delta$.

Space Complexity. A Generalized Morris counter tracking up to m with variance parameter $\gamma = \Theta(\varepsilon^2/k^2)$ uses $O(\log(1/\gamma) + \log\log m) = O(\log(k/\varepsilon) + \log\log m)$ bits. Maintaining $2RT = O(\frac{1}{\varepsilon^2}\log\frac{1}{\delta})$ such counters thus uses space $O\left(\frac{1}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\left(\log\frac{k}{\varepsilon} + \log\log m\right)\right)$. Additionally, the T initial pairwise independent hash functions $h_r : [n] \to [M]$ require $O(\log(1/\delta)\log n)$ additional bits of space. Similarly, the AMS hash functions σ require $O(\log M) = O(\log(k/\varepsilon))$ bits to generate. Altogether, the space complexity is $O\left(\frac{1}{\varepsilon^2}\log\left(\frac{1}{\delta}\right)\left(\log\frac{k}{\varepsilon} + \log\log m\right) + \log n \cdot \log\frac{1}{\delta}\right)$. ◀

References

1 Noga Alon, Phillip B. Gibbons, Yossi Matias, and Mario Szegedy. Tracking join and self-join sizes in limited storage. *J. Comput. Syst. Sci.*, 64(3):719–747, 2002. doi:10.1006/JCSS.2001.1813.

2 Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999. doi:10.1006/JCSS.1997.1545.

3 Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *J. Comput. Syst. Sci.*, 68(4):702–732, 2004. doi:10.1016/J.JCSS.2003.11.006.

4 Omri Ben-Eliezer, Talya Eden, and Krzysztof Onak. Adversarially robust streaming via dense-sparse trade-offs. In *5th Symposium on Simplicity in Algorithms, SOSA@SODA*, pages 214–227, 2022. doi:10.1137/1.9781611977066.15.

5 Jaroslaw Blasiok. Optimal streaming and tracking distinct elements with high probability. *ACM Trans. Algorithms*, 16(1):3:1–3:28, 2020. doi:10.1145/3309193.

6 Jaroslaw Blasiok, Jian Ding, and Jelani Nelson. Continuous monitoring of ℓ_p norms in data streams. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, volume 81, pages 32:1–32:13, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.32.

7 Mark Braverman and Or Zamir. Optimality of frequency moment estimation. In Michal Koucký and Nikhil Bansal, editors, *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC*, pages 360–370, 2025. doi:10.1145/3717823.3718171.

8 Vladimir Braverman, Jonathan Katzman, Charles Seidell, and Gregory Vorsanger. An optimal algorithm for large frequency moments using $O(n^{1-2/k})$ bits. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pages 531–544, 2014. doi:10.4230/LIPIcs.APPROX-RANDOM.2014.531.

9 Vladimir Braverman and Rafail Ostrovsky. Approximating large frequency moments with pick-and-drop sampling. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 16th International Workshop, APPROX, and 17th International Workshop, RANDOM. Proceedings*, 2013.

10 Vladimir Braverman, Emanuele Viola, David P. Woodruff, and Lin F. Yang. Revisiting frequency moment estimation in random order streams. In *45th International Colloquium on Automata, Languages, and Programming, ICALP*, pages 25:1–25:14, 2018. doi:10.4230/LIPIcs.ICALP.2018.25.

11 Amit Chakrabarti, Subhash Khot, and Xiaodong Sun. Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In *18th Annual IEEE Conference on Computational Complexity*, pages 107–117, 2003. doi:10.1109/CCC.2003.1214414.

12 Sumit Ganguly. Polynomial estimators for high frequency moments. *CoRR*, abs/1104.4552, 2011. arXiv:1104.4552.

13 Sumit Ganguly and David P. Woodruff. High probability frequency moment sketches. In *45th International Colloquium on Automata, Languages, and Programming, ICALP*, 2018. doi:10.4230/LIPIcs.ICALP.2018.58.

14 Anna C. Gilbert, Sudipto Guha, Piotr Indyk, Yannis Kotidis, S. Muthukrishnan, and Martin Strauss. Fast, small-space algorithms for approximate histogram maintenance. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing*, pages 389–398, 2002. doi:10.1145/509907.509966.

15 Elena Gribelyuk, Honghao Lin, David P. Woodruff, Huacheng Yu, and Samson Zhou. Lifting linear sketches: Optimal bounds and adversarial robustness. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC*, pages 395–406, 2025. doi:10.1145/3717823.3718227.

16 Elena Gribelyuk, Honghao Lin, David P. Woodruff, Huacheng Yu, and Samson Zhou. Adversarial robustness on insertion-deletion streams. In *Proceedings of the 58th Annual ACM Symposium on Theory of Computing, STOC*, pages 2278–2289, 2026. doi:10.1145/3798129.3800929.

17 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, 2006. doi:10.1145/1147954.1147955.

18 Piotr Indyk and David P. Woodruff. Optimal approximations of the frequency moments of data streams. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 202–208, 2005. doi:10.1145/1060590.1060621.

19 Rajesh Jayaram, David P. Woodruff, and Samson Zhou. Streaming algorithms with few state changes. *Proc. ACM Manag. Data*, 2(2):82, 2024. doi:10.1145/3651145.

20 Akshay Kamath, Eric Price, and David P. Woodruff. A simple proof of a new set disjointness with applications to data streams. In Valentine Kabanets, editor, *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*, volume 200 of *LIPIcs*, pages 37:1–37:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CCC.2021.37.

21 Daniel M. Kane, Jelani Nelson, Ely Porat, and David P. Woodruff. Fast moment estimation in data streams in optimal space. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC*, pages 745–754, 2011. doi:10.1145/1993636.1993735.

22 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. On the exact space complexity of sketching and streaming small norms. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 1161–1178, 2010. doi:10.1137/1.9781611973075.93.

23 Daniel M Kane, Jelani Nelson, and David P Woodruff. An optimal algorithm for the distinct elements problem. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 41–52, 2010. doi:10.1145/1807085.1807094.

24 Balachander Krishnamurthy, Subhabrata Sen, Yin Zhang, and Yan Chen. Sketch-based change detection: methods, evaluation, and applications. In *Proceedings of the 3rd ACM SIGCOMM Internet Measurement Conference, IMC*, pages 234–247, 2003. doi:10.1145/948205.948236.

25 Ping Li. Estimators and tail bounds for dimension reduction in $\ell_\alpha (0 < \alpha \leq 2)$ using stable random projections. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 10–19, 2008.

26 Honghao Lin, Zhao Song, David P. Woodruff, Shenghao Xie, and Samson Zhou. l_p sampling in distributed data streams with applications to adversarial robustness. In *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 4342–4409, 2026. doi:10.1137/1.9781611978971.160.

27 Robert Morris. Counting large numbers of events in small registers. *Communications of the ACM*, 21(10):840–842, 1978. doi:10.1145/359619.359627.

28 Shanmugavelayutham Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends® in Theoretical Computer Science*, 1(2):117–236, 2005. doi:10.1561/0400000002.

29 David P. Woodruff. Optimal space lower bounds for all frequency moments. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 167–175, 2004. URL: http://dl.acm.org/citation.cfm?id=982792.982817.

30 David P Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014. doi:10.1561/0400000060.

31 David P. Woodruff and Samson Zhou. Separations for estimating large frequency moments on data streams. In *48th International Colloquium on Automata, Languages, and Programming, ICALP*, pages 112:1–112:21, 2021. doi:10.4230/LIPIcs.ICALP.2021.112.

32 David P. Woodruff and Samson Zhou. Tight bounds for adversarially robust streams and sliding windows via difference estimators. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 1183–1196, 2021. doi:10.1109/FOCS52979.2021.00116.

33 David P. Woodruff and Samson Zhou. Adversarially robust dense-sparse tradeoffs via heavy-hitters. In *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems, NeurIPS*, 2024. doi:10.52202/079017-0348.

Approximate Cauchy-Schwarz Inequality and Improved Bounds for Sherali-Adams Refutation of Semirandom CSPs

Pravesh K. Kothari
Department of Computer Science, Princeton University, NJ, USA

Andrew D. Lin
Program in Applied and Computational Mathematics, Princeton University, NJ, USA

Abstract

We formulate an *approximate Cauchy-Schwarz* inequality and show that it is satisfied by solutions to the Sherali-Adams linear programming hierarchy (interpreted as "pseudo-distributions"). As a consequence, we resolve a question left open by the work of O'Donnell and Schramm [13] that they had explicitly attributed to the lack of such an inequality.

A Cauchy-Schwarz inequality is *exactly* satisfied by pseudo-distributions satisfying the constraints of the *sum-of-squares* semidefinite programming hierarchy and already has scores of applications. However, the proof there requires *global* positive semidefiniteness. Our approximate version, on the other hand, relies only on *local* positive semidefiniteness satisfied by the Sherali-Adams pseudo-distributions. Our formulation loses an additive error that scales with the L1 norm of the coefficients of the constituent polynomials, and this loss is asymptotically tight. Our proof is elementary and relies on a simple sampling argument.

As an application, we resolve a question left open in the work of O'Donnell and Schramm that gives a trade-off between constraint density and the Sherali-Adams degree for refuting random constraint satisfaction problems. Specifically, for odd arity CSPs, we show that the constraint density requirement for a given degree can be improved by a polynomial factor in n. Along the way, we observe that by a simple extension, the results in their work extend to a more general semirandom setting.

2012 ACM Subject Classification Theory of computation → Linear programming; Theory of computation → Approximation algorithms analysis

Keywords and phrases Sherali-Adams hierarchy, constraint satisfaction problems, semirandom CSPs, refutation, Cauchy-Schwarz inequality

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.77

Category RANDOM

1 Introduction

A degree d sum-of-squares (SoS) pseudo-distribution on $\{-1,1\}^n$ is a signed measure $D : \{-1,1\}^n \to \mathbb{R}$ such that the associated pseudo-expectation functional defined for any $f : \{-1,1\}^n \to \mathbb{R}$ by $\widetilde{\mathbb{E}}_D[f] = \sum_x D(x)f(x)$ satisfies 1) $\widetilde{\mathbb{E}}_D[1] = 1$ and 2) $\widetilde{\mathbb{E}}_D[f^2] \geqslant 0$ for every polynomial f of degree $\leqslant d/2$. Such pseudo-distributions are equivalent to feasible points for SDPs from the SoS semidefinite programming hierarchy. Degree d SoS pseudo-distributions are equivalent to *pseudo-moment matrices* – matrices M indexed by subsets of $[n]$ of size $\leqslant d/2$ whose entry at (S,T) is given by $\widetilde{\mathbb{E}}_D[\prod_{i \in S \Delta T} x_i]$. Pseudo-moment matrices are themselves in one-to-one correspondence with the set of all positive semidefinite matrices that satisfy some natural linear constraints[1]. The SoS semidefinite programming

[1] These constraints encode that $M(S,T)$ depends only on $S\Delta T$ and the diagonals of M are 1.

© Pravesh K. Kothari and Andrew D. Lin;
licensed under Creative Commons License CC-BY 4.0

hierarchy, at a high level, can be thought of as solving (in $n^{O(d)}$ time) the problem of finding a pseudo-expectation (equivalently, a pseudo-moment matrix) satisfying some additional linear constraints.

The analysis of algorithms (see the survey [7]) based on the SoS hierarchy requires reasoning about the pseudo-expectations of various polynomials arising in a given problem. Such analyses often rely on generalizing natural analytic inequalities that hold for expectations (w.r.t. probability distributions) to pseudo-expectations (w.r.t. pseudo-distributions). Indeed, the power of the pseudo-expectation view of SoS solutions comes largely from being able to "guess" true inequalities for pseudo-distributions by analogy with actual probability distributions.

The Cauchy-Schwarz Inequality

Perhaps the most fundamental of such general analytic inequalities is the Cauchy-Schwarz inequality:

$$\widetilde{\mathbb{E}}_D[fg] \leqslant \widetilde{\mathbb{E}}_D[f^2]^{1/2}\widetilde{\mathbb{E}}_D[g^2]^{1/2},$$

which is satisfied whenever D is a degree d pseudo-distribution and f, g are polynomials of degree $\leqslant d/2$. The Cauchy-Schwarz inequality appears in scores of applications of the sum-of-squares hierarchy, especially in executing the *proofs-to-algorithms* paradigm in algorithm design. The proof (and indeed, the truth) of the inequality relies on the positive semidefiniteness constraint that is often viewed as a *global* constraint on the pseudo-moments.

Such global positive semidefiniteness constraints are not satisfied by solutions to the weaker *Sherali-Adams* linear-programming hierarchy. An R-local Sherali-Adams solution can also be viewed as a degree $2R$ pseudo-distribution $D : \{-1,1\}^n \to \mathbb{R}$ with the key difference that the associated pseudo-expectation only satisfies a *local* version of (2). More precisely:

▶ **Definition 1.1** (R-local pseudo-distributions and pseudo-expectation)**.** *An R-local pseudo-distribution over the n-dimensional Boolean hypercube is a function $D : \{-1,1\}^n \to \mathbb{R}$ such that the associated pseudo-expectation defined by $\widetilde{\mathbb{E}}_D[f] := \sum_{x\in\{-1,1\}^n} D(x)f(x)$ satisfies:*

1. $\widetilde{\mathbb{E}}[1] = 1$,
2. *For every pointwise non-negative $f : \{-1,1\}^n \to \mathbb{R}$ such that f depends on only $\leqslant R$ out of n possible inputs, we have: $\widetilde{\mathbb{E}}[f] \geqslant 0$.*

Equivalently, for every non-negative function f that depends only on $\leqslant R$ variables, $\widetilde{\mathbb{E}}[f] \geqslant 0$. The corresponding pseudo-moment matrices now satisfy *local* positive semidefiniteness constraints – every principal submatrix that corresponds to pseudo-moments of any set of at most R variables is positive semidefinite. However, the matrix as a whole may not be PSD! This weakening allows encoding the constraints of a degree-d Sherali-Adams pseudo-expectation as an $n^{O(d)}$-size *linear* (as opposed to a semidefinite) program. The weakening to only satisfying local constraints comes at a cost. Results like the Cauchy-Schwarz inequality no longer hold, provably (see Theorem 2.2)!

Our goal in this work is to formulate an approximate version of the Cauchy-Schwarz inequality that can hold for Sherali-Adams pseudo-distributions. Before diving into our formulation, let us discuss our main motivation:

Sherali-Adams vs Sum-of-Squares

How does the power of the Sherali-Adams linear programming hierarchy compare with that of the SoS SDP hierarchy? On the one hand, lower bounds for constraint satisfaction problems such as Max-Cut [4] show that degree 2 SoS SDPs [9] can achieve drastically better

approximation ratios even as Sherali-Adams LPs of n^{ε}-degree fail to appreciably beat simply returning a uniformly random assignment. On the other hand, O'Donnell and Schramm [13] and the follow-up work due to Hopkins, Schramm, and Trevisan [11] show that degree n^c (for fixed $c < 1$) Sherali-Adams LPs can qualitatively match the performance of SDPs for Max-Cut! Perhaps more strikingly, they show that $O(1/\varepsilon)$-degree Sherali-Adams LPs (thus, in polynomial time!) can strongly refute (i.e., certify an upper bound of $1/2 + \varepsilon$ on the normalized value of) random k-XOR instances with $m \geqslant n^{k/2+\varepsilon}$ constraints **for even k**. This matches, up to n^{ε} factors for an arbitrarily small ε, the best-known refutation algorithms based on spectral methods [2, 3, 6].

The case of odd k

For random k-XOR refutation when k is odd, O'Donnell and Schramm prove a weaker bound of $m \geqslant n^{\lceil k/2 \rceil + \varepsilon}$ for the same guarantee that is off of the optimal $n^{k/2}$ by a fixed (i.e., that does not improve as $\varepsilon \to 0$) polynomial factor in n . This is because the argument in [13] essentially treats an odd k-XOR formula as a $(k+1)$-XOR and then applies their result for even arity XOR.

As an aside, such a gap between the case of even and odd ks has been a fixture in this line of work. The case of odd k, for which the above bound appears suboptimal, has indeed proved to be more difficult to tackle in all previous analyses [5, 2, 1, 12, 10] even with more powerful classes of spectral methods.

In the work of O'Donnell and Schramm [13], the polynomial factor "discrepancy" can be blamed on the lack of a Cauchy-Schwarz inequality for Sherali-Adams pseudo-distributions. Indeed, the authors themselves remark (see footnote on page 4): "*This is because we do not know how to relate the Sherali–Adams value of the objective function to its square (local versions of the Cauchy-Schwarz argument result in a loss). Such a relation would allow us to apply our techniques immediately to prove that Sherali Adams matches the SOS and spectral performance for odd as well as even k.*"

Our Results

In this work, we prove an approximate Cauchy-Schwarz inequality that holds for Sherali-Adams pseudo-distributions (so a "local" version as in the above quote) but with an additive loss that becomes negligible as the locality/degree of the Sherali-Adams LP increases. As an application, we get the right conjectured bound for refuting random k-XOR formulas for odd k. Along the way, we also observe a natural generalization of the ideas of [13] to the problem of semirandom k-XOR refutation where the set of tuples participating in constraints is arbitrary (i.e., worst-case) while the "right hand sides" are chosen independently at random.

Let us begin with our approximate Cauchy-Schwarz inequality. For any functions f, g, of $x = (x_1, x_2, \ldots, x_n)$ and any distribution D over x, $\mathbb{E}_{x\sim D}[f(x)g(x)] \leqslant \mathbb{E}[f(x)^2]^{1/2}\mathbb{E}[g(x)^2]^{1/2}$. The proof crucially relies on the positive semidefiniteness of the *moment* matrix of the pseudo-distribution. It provably does not hold for Sherali-Adams pseudo-distributions that only satisfy *local* positive semidefiniteness. Indeed, in Section 2.1, we observe the following seemingly drastic failure of the Cauchy-Schwarz inequality.

▶ **Theorem 1.2** (Sherali-Adams Cannot Satisfy Cauchy-Schwarz Exactly, see Theorem 2.2)**.** *For any R and $n \gg R$, there exists an R-local Sherali-Adams pseudo-distribution on $\{-1, 1\}^n$ such that for the linear polynomial $p(x) = \sum_i x_i$, we have: $\widetilde{\mathbb{E}}_D[p(x)^2] = 0$, and, $\widetilde{\mathbb{E}}_D[p] \geqslant (1 - o_R(1))\frac{n}{\sqrt{R-1}}$.*

Despite this failure, we show that there is a natural way to formulate a correct, approximate Cauchy-Schwarz inequality. To the best of our knowledge, such an approximate formulation has not appeared in prior works. Notice that the above counter-example rules out an approximate Cauchy-Schwarz inequality with a *multiplicative* error. Our formulation has a bound on the *additive* error instead.

▶ **Theorem 1.3** (Approximate Cauchy-Schwarz Inequality). *Let $f, g : \{-1,1\}^n \to \mathbb{R}$ be polynomials of degree $\leqslant d$. Let $\|f\|_1, \|g\|_1$ be the L1 norms of the coefficients of f, g, respectively. Then, for every R-local pseudo-distribution $\mathcal{D}$ on $\{-1,1\}^n$ and any positive integers s, t such that $d(s+t) \leqslant R$, we have:*

$$\left|\widetilde{\mathbb{E}}[fg]\right| \leqslant \sqrt{(1-\frac{1}{s})\widetilde{\mathbb{E}}[f^2] + \frac{1}{s}\|f\|_1^2}\sqrt{(1-\frac{1}{t})\widetilde{\mathbb{E}}[g^2] + \frac{1}{t}\|g\|_1^2}.$$

▶ Remark 1.4. Note that $\|f\|_1$ and $\|g\|_1$ are basis dependent quantities. This is to be expected as Sherali-Adams (unlike sum-of-squares) is not basis independent. Next, our bounds are written in the $\{\pm 1\}$ basis. They can, of course, be transformed into the $\{0,1\}$ basis, but the quantities appearing will change. Finally, we present the inequality with s, t as independent parameters (as opposed to setting them both to be $R/2$, say) since we anticipate that in applications, $\|f\|_1, \|g\|_1$ could be different and the freedom to modulate s, t accordingly may lead to better bounds.

As we observe in Section 2.1, Theorem 1.2 shows that the formulation we prove in Theorem 1.3 is tight down to the exact constant as $n/R \to \infty$.

Applications

We apply Approximate Cauchy-Schwarz to prove tight bounds for CSP refutation.

▶ **Theorem 1.5** (Sherali-Adams Refutation of Random and Semirandom k-XOR; see Theorems 3.4 and 4.2). *Suppose we have a random or semirandom k-XOR instance on n variables with $n^{k/2+\delta}$ expected (in the random case) or total (in the semirandom case) constraints. Then with high probability, there exists some $R = O_{\varepsilon,\delta,k}(1)$ such that for all R-local Sherali-Adams pseudodistributions D on $\{-1,1\}^n$, we have $\widetilde{\mathbb{E}}_D \psi(x) \leqslant 2\varepsilon|\mathcal{H}|$.*

In the case of odd k, we use our approximate Cauchy-Schwarz lemma to show that the standard spectral certificate used to refute random odd k-XOR instances in [2] implies a Sherali-Adams refutation, which allows us to refute with just $n^{k/2+\delta}$ constraints. In addition, we show that we can refute semirandom k-XOR in Sherali-Adams, matching the SOS algorithms from [1]. This was previously unknown for both even and odd k.

2 Sherali-Adams Proof of an Approximate Cauchy-Schwarz Inequality

In this section, we show that Sherali-Adams pseudo-distributions satisfy a natural approximate Cauchy-Schwarz inequality. In the next section, we will see how this inequality helps linear-programming-based refutations of random constraint satisfaction problems.

Let us formulate the approximate Cauchy-Schwarz inequality:

▶ **Theorem 2.1** (Approximate Cauchy-Schwarz Inequality). *Let $f, g : \{-1,1\}^n \to \mathbb{R}$ be polynomials of degree $\leqslant d$. Let $\|f\|_1, \|g\|_1$ be the L1 norms of the coefficients of f, g respectively when written in the standard monomial basis over $\{-1,1\}^n$. Then, for every R-local pseudo-distribution $\mathcal{D}$ on $\{-1,1\}^n$, and any positive integers s, t such that $d(s+t) \leqslant R$, we have:*

$$\left|\widetilde{\mathbb{E}}[fg]\right| \leqslant \sqrt{(1-\frac{1}{s})\widetilde{\mathbb{E}}[f^2] + \frac{1}{s}\|f\|_1^2}\sqrt{(1-\frac{1}{t})\widetilde{\mathbb{E}}[g^2] + \frac{1}{t}\|g\|_1^2}.$$

Our proof is based on a simple sampling scheme that allows us to write f and g as averages over *local* functions with some additive "sampling" error.

Proof. We present the proof for $d = 1$. The case of larger d is the same but, unlike below, where we "sample" monomials of degree 1, we sample monomials of degree $\leqslant d$ instead.

Let $f(x) = a_0 + \sum_{i \leqslant n} a_i x_i$ and $g(x) = b_0 + \sum_i b_i x_i$. We will assume that $0 < \|f\|_1 = \sum_i |a_i|$ and $0 < \|g\|_1 = \sum_i |b_i|$ (if either of these quantities is 0 then there is nothing to prove).

Let us sample an unbiased estimator for f. Define a distribution p on $\{0, 1, 2, \cdots, n\}$ by setting $p_i = \frac{|a_i|}{\|f\|_1}$. For each $i \in \{0, 1, 2, \cdots, n\}$, let $\chi_i(x) = x_i$ if $i > 0$ and $\chi_0(x) = 1$. Draw $i_1, i_2, \ldots, i_s$ independently from p and define:

$$F = \frac{1}{s}\sum_{\ell=1}^{s} \frac{a_{i_\ell}}{p_{i_\ell}}\chi_{i_\ell}(x).$$

Then, clearly, F depends on at most s variables. Further, $\mathbb{E}[F] = f$, where the expectation is with respect to p. Similarly, construct G from g using t samples.

Now, let $\mathcal{D}$ be any R-local pseudo-distribution on $x_1, x_2, \ldots, x_n \in \{\pm 1\}$. Since F and G are s, t-local respectively and $s + t \leqslant R$, we must have:

$$|\widetilde{\mathbb{E}}_D[FG]| \leqslant \widetilde{\mathbb{E}}_D[F^2]^{1/2}\widetilde{\mathbb{E}}_D[G^2]^{1/2}. \tag{1}$$

Now, since F and G are sampled independently, we also have:

$$\mathbb{E}_{\text{sampling}}\widetilde{\mathbb{E}}[FG] = \widetilde{\mathbb{E}}[fg].$$

Thus, using (1), taking expectations with respect to the sampling process and applying Jensen's inequality for the expectation of the square root function gives

$$\left|\widetilde{\mathbb{E}}[fg]\right| \leqslant \mathbb{E}_{\text{sampling}}\widetilde{\mathbb{E}}_{\mathcal{D}}[F^2]^{1/2}\widetilde{\mathbb{E}}_{\mathcal{D}}[G^2]^{1/2} \leqslant \sqrt{(\mathbb{E}_{\text{sampling}}\widetilde{\mathbb{E}}_{\mathcal{D}}[F^2])}\sqrt{\mathbb{E}_{\text{sampling}}\widetilde{\mathbb{E}}_{\mathcal{D}}[G^2]}. \tag{2}$$

We will now finish the proof by computing the RHS. Let $X_\ell = \frac{a_{i_\ell}}{p_{i_\ell}}\chi_{i_\ell}(x)$, so $F = \frac{1}{s}\sum_{\ell=1}^{s} X_\ell$.

For two distinct samples $\ell \neq m$, we have:

$$\mathbb{E}_{i_\ell, i_m}[\widetilde{\mathbb{E}}[X_\ell X_m]] = \widetilde{\mathbb{E}}[f^2].$$

On the other hand, when $\ell = m$, we have:

$$\mathbb{E}_{i_\ell}[\widetilde{\mathbb{E}}[X_\ell^2]] = \sum_i \frac{a_i^2}{p_i} = (\sum_i |a_i|)^2 = \|f\|_1^2.$$

Thus,

$$\mathbb{E}_{\mathsf{sampling}}[\widetilde{\mathbb{E}}[F^2]] = \frac{s(s-1)}{s^2}\widetilde{\mathbb{E}}[f^2] + \frac{s}{s^2}\|f\|_1^2 = (1-1/s)\widetilde{\mathbb{E}}[f^2] + \frac{\|f\|_1^2}{s}.$$

A similar argument for G gives:

$$\mathbb{E}_{\mathsf{sampling}}[\widetilde{\mathbb{E}}[G^2]] = (1-1/t)\widetilde{\mathbb{E}}[g^2] + \frac{\|g\|_1^2}{t}.$$

Plugging back in (2) finishes the proof. ◀

2.1 A Matching Lower Bound

In this section, we prove that the additive error appearing in our approximate Cauchy-Schwarz inequality is necessary and, in fact, that our bound is quantitatively optimal.

▶ **Theorem 2.2.** *Let R, n be such that $n/R \to \infty$. Then, there exists an R-local Sherali-Adams pseudo-distribution on $\{-1,1\}^n$ such that for the linear polynomial $p(x) = \sum_i x_i$, we have: $\widetilde{\mathbb{E}}_D[p(x)^2] = 0$, and,*

$$\widetilde{\mathbb{E}}_D[p] \geqslant (1 - O(1/R + R/n))\frac{n}{\sqrt{R-1}}.$$

▶ Remark 2.3. Applying Theorem 2.1 with $f = p$, $g = 1$, $s = R-1$ and $t = 1$ gives $\widetilde{\mathbb{E}}_D[p] \leqslant n/\sqrt{R-1}$. Thus, the additive error in Theorem 2.1 is asymptotically tight with the correct leading constant. Notice also that this example rules out an approximate Cauchy-Schwarz inequality with any finite *multiplicative* error.

Proof. We will define the pseudo-distribution D by describing a local distribution on every collection T of R variables. The local distributions will be permutation-invariant on T and thus, the marginal distribution on any $T' \subseteq T$ will depend only on $|T'|$. Thus, in particular, local consistency holds: for any $T' \subseteq T_1 \cap T_2$ for two different sets of size R each, the marginal of the local distributions on T_1 and T_2 on T' agree. This implies immediately the existence of an R-local Sherali-Adams pseudo-distribution with pseudo-moments agreeing with the local distributions we will construct.

For every $T \subseteq [n]$ of size exactly R, let $Y_T = \sum_{i \in T} x_i$. We will sample Y_T from a distribution on $[-R, R]$ and then set the constituent x_is to be uniform conditioned on satisfying $\sum_i x_i$ being equal to the sampled value of Y_T. This will ensure permutation-invariance that we desire immediately.

What remains to be described is the specific distribution to sample the random variable Y_T on $[-R, R]$. We will choose this carefully to have prescribed first and second moments (which will govern $\widetilde{\mathbb{E}}_D[p]$ and $\widetilde{\mathbb{E}}_D[p^2]$ respectively).

Let $A = \frac{R(n-R)}{n-1}$. This will be our prescribed 2nd moment for Y_T. Notice that $A = R(1 - O(R/n))$ and $\sqrt{A} = \sqrt{R}(1 - O(R/n))$. Choose a to be the largest positive integer in $\{-R, -R+2, \cdots, R-2, R\}$ such that $a \leqslant \sqrt{A}$. Set $b = a + 2$. Choose θ so that $(1-\theta)a^2 + \theta b^2 = A$, that is, set $\theta = \frac{A - a^2}{b^2 - a^2}$. Observe that since $a^2 \leqslant A \leqslant b^2$, $\theta \in [0, 1]$.

Now, consider the following distribution for Y_T. With probability $1 - \theta$, output a, with probability θ, output b. Choose $x_T \in \{-1, 1\}^T$ to be uniform over all assignments satisfying $\sum_i x_i = Y_T$.

Let's compute $\widetilde{\mathbb{E}}_D[x_i x_j]$ for $i \neq j$. Observe that $Y_T^2 = R + 2\sum_{i<j, i,j \in T} x_i x_j$. Since $\mathbb{E}[Y_T^2] = A$, we must have $\widetilde{\mathbb{E}}_D[x_i x_j] = \frac{(A-R)}{R(R-1)} = -\frac{1}{n-1}$. Thus, $\widetilde{\mathbb{E}}_D[p^2] = 0$.

Next, a simple computation gives $\widetilde{\mathbb{E}}_D[x_i] = 1/R\widetilde{\mathbb{E}}_D[Y_T] = 1/R[(1-\theta)a + \theta b] = \frac{1}{\sqrt{R}}(1 - O(1/R + R/n))$.

Thus, $\widetilde{\mathbb{E}}_D[p] = (1 - O(1/R + R/n))\frac{n}{\sqrt{R}} = (1 - O(1/R + R/n))\frac{n}{\sqrt{R-1}}$. ◀

3 Refuting Random k-XOR

In this section, we will show that Sherali-Adams can refute random k-XOR instances, which we define below, as an application of approximate Cauchy-Schwarz.

▶ **Definition 3.1.** *Let $k \geqslant 2$ be an integer. We say that ψ is a random k-XOR instance with pn^k expected clauses if for all $U \in [n]^k$, we add U to $\mathcal{H}$ with probability p and sample an independent Rademacher (± 1) sign b_U. Then we write $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$, where $x^U \coloneqq \prod_{i\in U} x_i$.*

▶ Remark 3.2. Notice that if an $\frac{1}{2} + \gamma$ fraction of constraints are satisfied, i.e. we have $b_U \cdot x^U = 1$ for $\frac{1}{2} + \gamma$ fraction of $U \in \mathcal{H}$, then $\psi(x) = 2\gamma|\mathcal{H}|$. Therefore, certifying that we cannot satisfy a large fraction of constraints is equivalent to certifying an upper bound on $\psi(x)$.

We also define the following notation that we will use:

▶ **Definition 3.3.** *Let $\psi = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a k-XOR instance with clause set $\mathcal{H}$. For all $U \notin \mathcal{H}$, we let $b_U = 0$. Define $\mathcal{H}_i \coloneqq \{C \in \mathcal{H} : C_k = i\}$ be the subset of clauses in $\mathcal{H}$ whose last coordinate is i.*

We will prove the following theorem:

▶ **Theorem 3.4.** *Let k be an odd integer and $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a random k-XOR instance with $n^{k/2+\delta}$ expected constraints. Suppose we have $2^{O(k)} \log^3 n \leqslant n^{\delta/2}$. Then for all $\Omega(1) = \gamma \leqslant 1/2$, there exists some $R = O_{\gamma,\delta,k}(1)$ such that every R-local pseudodistribution $\mathcal{D}$ satisfies $\widetilde{\mathbb{E}}_{\mathcal{D}}\psi(x) \leqslant 2\gamma|\mathcal{H}|$.*

▶ Remark 3.5. Using a standard Fourier analytic reduction, we can generalize this result to general k-CSPs.

The state-of-the-art refutation algorithms for random (and semirandom) k-XOR instances are based on spectral certificates. Informally, the idea is to show that the "value" polynomial of any k-XOR instance (the polynomial that computes the advantage over $1/2$ of an assignment) can be upper bounded at every x by a quadratic form of a certain matrix built from the instance. One then establishes that when the input instance is random (or semirandom) with appropriately many constraints, then, the spectral norm of this matrix is small. We will call a matrix satisfying sufficient conditions for obtaining a Sherali-Adams refutations a spectral certificate.

▶ **Definition 3.6** (Spectral Certificates)**.** *Let k be an odd integer and $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a k-XOR instance. Define $A(\psi)$ to be a $n^{k-1} \times n^{k-1}$ matrix defined as follows: For all $(S_1, T_1) \neq (S_2, T_2) \in [n]^{(k-1)/2} \times [n]^{(k-1)/2}$, we have*

$$A(\psi)_{(S_1,S_2),(T_1,T_2)} \coloneqq \sum_{i\in[n]} b_{S_1T_1i}b_{S_2T_2i} + b_{T_1S_1i}b_{T_2S_2i}\,,$$

and $A(\psi)_{(S_1,S_1),(S_2,S_2)} = 0$. *We call* $A(\psi)$ *a* δ-spectral certificate for $\mathcal{H}$ *if*

$$\|A(\psi)\| \leqslant n^{-\delta} \left(\min_{(S_1,S_2)} \left\|A(\psi)_{(S_1,S_2)}\right\|_1 \right) .$$

We let $D(\psi)_{S_1,S_2} \coloneqq \|A_{S_1,S_2}\|_1$ *and write* $D(\psi) \coloneqq \operatorname{diag}(\{D(\psi)_{S_1,S_2}\})$. *Observe that*

$$x^{\odot(k-1)} A(\psi) x^{\odot(k-1)} = 2 \sum_i \sum_{U \neq U' \in \mathcal{H}_i} b_U b_{U'} x^U x^{U'} .$$

With high probability over the choice of ψ, the matrix $A(\psi)$ is a spectral certificate. The proof of this is simple and follows from the work of [2], and is deferred to the appendix.

▶ **Lemma 3.7.** *Suppose* ψ *is a random* k*-XOR instance with* $n^{k/2+\delta}$ *constraints such that* $2^{O(k)} \log^3 n \leqslant n^{\delta/2}$. *Then* $A(\psi)$ *is a* $\frac{\delta}{2}$*-spectral certificate for* ψ *with high probability.*

One can view the main idea of O'Donnell and Schramm's work as showing (under appropriate conditions) that spectral bounds on certifying matrices such as above can be proven using the Sherali-Adams linear program. We will abstract their analysis into the following lemma:

▶ **Lemma 3.8** (Abstracting out [13])**.** *Let* A *be a symmetric matrix with integer entries with rows and columns indexed by* t*-tuples of* $[n]$*, and let* D *be its degree matrix. Let* $\pi_* = \frac{1}{\sum_{S,T} |A_{S,T}|} \min_S \|A_S\|_1$, $\rho(\cdot)$ *denote the spectral radius of a matrix, and* $\eta > \min(\pi_*^{-1/2}, \rho(D^{-1}A))$. *Take* $\ell \coloneqq \left\lceil \frac{1}{4} \frac{\log(\eta^2 \pi_*)}{\log(\rho(D^{-1}A)/\eta)} \right\rceil$ *and* $p \coloneqq \lceil (\frac{2}{5\eta})^{2\ell} \rceil$. *Then for* $R(A) \coloneqq tp\ell + t$*, any* R*-local pseudoexpectation* $\mathcal{D}$ *on* $x_1, \cdots, x_n$ *satisfies*

$$\left| \widetilde{\mathbb{E}}_{\mathcal{D}} (x^{\odot t})^\top A x^{\odot t} \right| \leqslant \eta \sum_{S,T} |A_{S,T}|,$$

where $x^{\odot t}$ *is the* n^t*-dimensional vector whose entries are indexed by* t*-tuples of* $[n]$ *and are equal to the product of the corresponding entries.*

▶ **Lemma 3.9.** *Let* k *be an odd integer and* $\psi(x) = \sum_{U \in \mathcal{H}} b_U \cdot x^U$ *be a* k*-XOR instance such that* $|\mathcal{H}| \geqslant \frac{36n}{\gamma^2}$ *and* $|\mathcal{H}_i| \leqslant \frac{|\mathcal{H}|}{n}(1+\gamma)$ *for all* i. *Suppose that* $A(\psi)$ *is a* δ*-spectral certificate for* ψ. *Then for all* $\Omega(1) = \gamma \leqslant \frac{1}{2}$, $R = O_{\gamma,\delta,k}(1)$*-rounds of Sherali-Adams can certify that* $\psi(x) \leqslant 2\gamma|\mathcal{H}|$*; in other words, for all* R*-local pseudodistributions* $\mathcal{D}$, $|\widetilde{\mathbb{E}}_{\mathcal{D}} \psi(x)| \leqslant 2\gamma|\mathcal{H}|$.

Proof. Let $\psi_i = \sum_{U \in \mathcal{H}_i} b_U x^U$, $R \coloneqq \max\{R(A(\psi)), \frac{10k(1+\gamma)^2}{\gamma^2}\} = O_{\gamma,\delta,k}(1)$, where $R(A(\psi))$ is the quantity from Lemma 3.8 when $\eta = \gamma^2$, and $\widetilde{\mathbb{E}}$ be any R-local pseudodistribution. A δ-spectral certificate $A(\psi)$ satisfies $\rho(D^{-1}(\psi)A(\psi)) \leqslant n^{-\delta}$ (recall that $\rho(\cdot)$ is the spectral radius of a matrix) and therefore by Lemma 3.8,

$$\begin{aligned} \widetilde{\mathbb{E}} \sum_i \psi_i^2(x) &= \widetilde{\mathbb{E}} \sum_{U \in \mathcal{H}} b_U^2 + \frac{1}{2} \widetilde{\mathbb{E}} x^{\odot(k-1)} A(\psi) x^{\odot(k-1)} \\ &\leqslant |\mathcal{H}| + \frac{\gamma^2}{2} \sum_i |\mathcal{H}_i|(|\mathcal{H}_i| - 1) \leqslant |\mathcal{H}| + \frac{\gamma^2}{2} \frac{|\mathcal{H}|^2}{n} (1+\gamma)^2 , \end{aligned}$$

so by Theorem 2.1, we have for $s \geqslant \frac{9(1+\gamma)^2}{\gamma^2}$,

$$\left|\widetilde{\mathbb{E}}\psi(x)\right| = \left|\widetilde{\mathbb{E}}\sum_i \psi_i(x)\right| \leqslant \sum_i \sqrt{\left(1-\frac{1}{s}\right)\widetilde{\mathbb{E}}\psi_i^2 + \frac{1}{s}\|\psi_i\|_1^2}$$
$$\leqslant \sqrt{n\left(\left(1-\frac{1}{s}\right)\sum_i \widetilde{\mathbb{E}}\psi_i^2 + \frac{1}{s}\sum_i |\mathcal{H}_i|^2\right)}$$
$$\leqslant \sqrt{n|\mathcal{H}|} + \gamma(1+\gamma)|\mathcal{H}| + \frac{|\mathcal{H}|}{\sqrt{s}}(1+\gamma) \leqslant 2\gamma|\mathcal{H}|\,. \quad \blacktriangleleft$$

Proof of Theorem 3.4. For random k-XOR, Chernoff bounds imply that $|\mathcal{H}_i| \leqslant \frac{|\mathcal{H}|}{n}(1+\gamma)$ for all i with high probability. Then the result follows directly from Lemma 3.7 and Lemma 3.9. $\blacktriangleleft$

4 Refuting Semirandom k-XOR

In this section, we show that Sherali-Adams can refute semirandom instances of k-XOR.

▶ **Definition 4.1.** *Let k be an integer and $\mathcal{H} \subseteq [n]^k$. We say that ψ is a semirandom k-XOR instance with clause set $\mathcal{H}$ if for $U \in \mathcal{H}$, we sample an independent Rademacher (± 1) sign b_U and write $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$.*

▶ **Theorem 4.2.** *Let k be an integer. $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a semirandom k-XOR instance with clause set $\mathcal{H}$ such that $|\mathcal{H}| = n^{k/2+\delta}$ total constraints. Suppose we have $\gamma \geqslant n^{-\delta/4}4\sqrt{15}$ and $16384k^4\log^3 n \leqslant n^{\delta/18}$. Then for all $\Omega(1) = \gamma \leqslant 1/2$, with high probability, there exists some $R = O_{\gamma,\delta,k}(1)$ such that every R-local pseudodistribution $\mathcal{D}$ satisfies $\widetilde{\mathbb{E}}_{\mathcal{D}}\psi(x) \leqslant 2\gamma|\mathcal{H}|$.*

For semirandom odd k-XOR, the $\mathcal{H}_i$ can be much larger than $\frac{|\mathcal{H}|}{n}$, in which case we need to partition large $\mathcal{H}_i$ into smaller sets to ensure the matrix $A(\psi)$ does not have too many nonzero entries. In addition, we can no longer guarantee that $A(\psi)$ is a spectral certificate with high probability. We thus define a notion of a regular partition and a refined notion of a spectral certificate that allows us to delete a negligible fraction of rows and columns of the matrix.

▶ **Definition 4.3.** *For all $\mathcal{H}_1 \sqcup \cdots \sqcup \mathcal{H}_n$, Let $\mathcal{H} \subseteq [n]^k$. We call a partition $P \coloneqq \bigsqcup_i P_i \coloneqq \bigsqcup_i \bigsqcup_{j\in[p_i]} \mathcal{H}_i^{(j)}$ of the $\mathcal{H}_i$s* regular *if it has a total of at most $2n$ parts and $|\mathcal{H}_i^{(j)}| \leqslant \frac{|\mathcal{H}|}{n}$ for all i, j.*

▶ **Definition 4.4** ((γ,δ)-Spectral Certificates)**.** *Let k be an odd integer and $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a k-XOR instance with clause set $\mathcal{H}$. Suppose we have a partition $P_i \coloneqq \sqcup_{j\in[p_i]}\mathcal{H}_i^{(j)}$ of each $\mathcal{H}_i$, and let $P = (P_1, \cdots, P_n)$. Define $A'_P(\psi)$ to be a $n^{k-1} \times n^{k-1}$ matrix defined as follows: For all $(S_1, T_1) \neq (S_2, T_2) \in [n]^{(k-1)/2} \times [n]^{(k-1)/2}$, we have*

$$A'_P(\psi)_{(S_1,S_2),(T_1,T_2)} \coloneqq \sum_{i\in[n]}\sum_{j\in[p_i]} b_{S_1T_1i}b_{S_2T_2i} \cdot \mathbf{1}\{S_1T_1i, S_2T_2i \in \mathcal{H}_i^{(j)}\}$$
$$+ b_{T_1S_1i}b_{T_2S_2i} \cdot \mathbf{1}\{T_1S_1i, T_2S_2i \in \mathcal{H}_i^{(j)}\}\,,$$

*and $A'_P(\psi)_{(S_1,S_1),(S_2,S_2)} = 0$. Let $A(\psi)$ be a principal submatrix of $A'_P(\psi)$, and define $D(\psi)_{S_1,S_2} \coloneqq \|A_{S_1,S_2}\|_1$ and write $D(\psi) \coloneqq \operatorname{diag}(\{D(\psi)_{S_1,S_2}\})$. We call $A(\psi)$ a (γ,δ)-*spectral certificate of ψ with respect to P *if the following hold:*

- *$A(\psi)$ can be formed by removing a collection of rows of $A'_P(\psi)$ and their corresponding columns whose ℓ_1 norms sum to at most $\frac{\gamma^2}{2}\sum_{i\in[n]}\sum_{j\in[p_i]}|\mathcal{H}_i^{(j)}|(|\mathcal{H}_i^{(j)}|-1)$,*
- *$\rho(D^{-1}(\psi)A(\psi)) \leqslant n^{-\delta/2}$, where $\rho(\cdot)$ denotes the spectral radius of a matrix.*

Observe that (abusing notation and treating $A(\psi)$ as a $n^{k-1}\times n^{k-1}$ matrix where the removed rows are replaced by zeros), we have

$$x^{\odot(k-1)}A'_P(\psi)x^{\odot(k-1)} = 2\sum_i \sum_{j\in[p_i]} \sum_{U\neq U'\in\mathcal{H}_i^{(j)}} b_U b_{U'} x^U x^{U'},$$

and

$$x^{\odot(k-1)}\left(A'_P(\psi)-A(\psi)\right)x^{\odot(k-1)} \leqslant \frac{\gamma^2}{2}\sum_{S_1,S_2}|A'_P(\psi)_{S_1,S_2}|.$$

▶ **Definition 4.5.** *Let k be an integer and $\psi(x)=\sum_{U\in\mathcal{H}} b_U\cdot x^U$ be a k-XOR instance with clause set $\mathcal{H}$. Then we define $\psi^{\mathbf{1}}(x) \coloneqq \sum_{U\in\mathcal{H}} x^U$ to be the k-XOR instance with the same constraints but all ones on the RHS.*

▶ Remark 4.6. Note that any entry $A'_P(\psi)_{(S_1,S_2),(T_1,T_2)}$ is the sum of $A'_P(\psi^{\mathbf{1}})_{(S_1,S_2),(T_1,T_2)}$ independent Rademacher random variables. In particular, if $A'_P(\psi^{\mathbf{1}})_{(S_1,S_2),(T_1,T_2)}\neq 0$, then $A'_P(\psi)_{(S_1,S_2),(T_1,T_2)}=0$ with probability at most $\frac{1}{2}$.

We will show that the existence of a (γ,δ)-spectral certificate of ψ implies that Sherali-Adams admits a refutation of ψ.

▶ **Lemma 4.7.** *Let k be an odd integer and $\psi(x)=\sum_{U\in\mathcal{H}} b_U\cdot x^U$ be a semirandom k-XOR instance. Suppose that there exists a (γ,δ)-spectral certificate for ψ with respect to a regular partition P of all $\mathcal{H}_i$. Then if $\gamma\leqslant\frac{1}{2}$ and $|\mathcal{H}|\geqslant\frac{4n}{\gamma^2}$, then $R=O_{\gamma,\delta,k}(1)$-rounds of Sherali-Adams can certify that $\psi(x)\leqslant 2\gamma|\mathcal{H}|$; in other words, for all R-local pseudodistributions $\mathcal{D}$, $|\widetilde{\mathbb{E}}_{\mathcal{D}}\psi(x)|\leqslant 2\gamma|\mathcal{H}|$.*

Proof of Lemma 4.7. Let $\psi_{ij}=\sum_{U\in\mathcal{H}_i^{(j)}} b_U x^U$ and $A(\psi)$ be a (γ,δ)-spectral certificate of ψ. Let $R\coloneqq\max\{R(A(\psi)),\frac{5k}{\gamma^2}\}=O_{\gamma,\delta,k}(1)$, where $R(A(\psi))$ is the quantity from Lemma 3.8 when $\eta=\frac{\gamma^2}{2}$, and $\widetilde{\mathbb{E}}$ be an R-local pseudodistribution. Then $\rho(D^{-1}(\psi)A(\psi))\leqslant n^{-\delta/2}$ and thus by Lemma 3.8,

$$\begin{aligned}
\widetilde{\mathbb{E}}\sum_i\sum_{j\in[p_i]}\psi_{ij}^2(x) &= \widetilde{\mathbb{E}}\sum_{U\in\mathcal{H}} b_U^2 + \frac{1}{2}\widetilde{\mathbb{E}}x^{\odot(k-1)}A'(\psi)x^{\odot(k-1)}\\
&\leqslant |\mathcal{H}| + \frac{1}{2}\widetilde{\mathbb{E}}x^{\odot(k-1)}A(\psi)x^{\odot(k-1)} + \frac{1}{2}\widetilde{\mathbb{E}}x^{\odot(k-1)}(A'(\psi)-A(\psi))x^{\odot(k-1)}\\
&\leqslant |\mathcal{H}| + \left(\frac{\gamma^2}{2}+\frac{\gamma^2}{4}\right)\sum_i\sum_{j\in[p_i]}|\mathcal{H}_i^{(j)}|(|\mathcal{H}_i^{(j)}|-1)\\
&\leqslant |\mathcal{H}| + \frac{3\gamma^2}{4}\frac{|\mathcal{H}|^2}{n},
\end{aligned}$$

where the last line uses the fact that our partition is regular and therefore each $\mathcal{H}_i^{(j)}$ has size at most $\frac{|\mathcal{H}|}{n}$. Thus by Theorem 2.1, since P has at most $2n$ parts, we have, for $s\geqslant\frac{4}{\gamma^2}$,

$$\left|\widetilde{\mathbb{E}}\psi\right| = \left|\widetilde{\mathbb{E}}\sum_i \sum_{j\in[p_i]} \psi_{ij}(x)\right| \leqslant \sum_i \sum_{j\in[p_i]} \sqrt{\left(1-\frac{1}{s}\right)\widetilde{\mathbb{E}}\psi_{ij}^2 + \frac{1}{s}\left\|\psi_{ij}\right\|_1^2}$$
$$\leqslant \sqrt{2n\left(\left(1-\frac{1}{s}\right)\sum_i\sum_{j\in[p_i]}\widetilde{\mathbb{E}}\psi_{ij}^2 + \frac{|\mathcal{H}_i|^2}{sn}\right)}$$
$$\leqslant \sqrt{2n|\mathcal{H}| + 3\gamma^2|\mathcal{H}|^2 + \frac{2|\mathcal{H}|^2}{s}} \leqslant 2\gamma|\mathcal{H}|\,. \qquad \blacktriangleleft$$

Finally, we note that given sufficient regularity conditions on $\mathcal{H}$, there exists a spectral certificate of ψ with high probability.

▶ **Lemma 4.8.** *Let $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a semirandom k-XOR instance with constraint set $\mathcal{H} \subseteq [n]^k$ such that $|\mathcal{H}| = n^{k/2+\delta}$ and let P be a regular partition of the $\mathcal{H}_i$s. Suppose $\mathcal{H}$ has the following properties:*

- *For all $S \in [n]^{(k-1)/2}$, $i \in [n]$, there are at most $n^{\delta/2}$ total clauses of the form STi or TSi in $\mathcal{H}$.*
- *For all $S_1T_1 \in [n]^{(k-1)/2} \times [n]^{(k-1)/2}$, there are at most $n^{\delta/2}$ total clauses of the form $S_1T_1i \in \mathcal{H}$.*

Suppose that $\gamma \geqslant n^{-\delta/4}\sqrt{60}$ and $16384k^4\log^3 n \leqslant n^{\delta/18}$. Then with high probability, there exists a $(\gamma, \delta/9)$-spectral certificate of ψ with respect to P.

We defer the proof of this lemma to Section 4.3.

4.1 Refuting induced 2-XOR instances

Unlike the random case, a semirandom XOR instance ψ does not necessarily have a spectral certificate: when large groups of variables occur together in a large number of clauses, the spectral norm of $A'(\psi)$ can be large and making it sufficiently small requires pruning a large (possible almost all) of the rows. Instead, we can partition our set of clauses $\mathcal{H}$ into two subparts $\mathcal{H}'$ and $\mathcal{H}''$, where for $\mathcal{H}'$ is the "approximately regular" part where we can construct the aforementioned spectral certificate, and for the second part $\mathcal{H}''$, we instead construct a 2-XOR instance that we can refute using Sherali-Adams. At a high level, this resembles the approach of [1], however, their method for refuting the 2-XOR instance for $\mathcal{H}''$ relies on bounding the value of the instance by an infinity-to-1 norm and using the fact that SOS can recognize Grothendieck's inequality. This is not known for Sherali-Adams, so we must handle this case differently.

We define the notion of an induced 2-XOR instance below. This is a 2-XOR instance formed by splitting each clause U of a k-XOR instance into two parts $S \sqcup T$, defining a variable y_S and y_T for each of the two parts, and replacing the k-XOR constraint $b_U x^U$ with the 2-XOR constraint $b_U y_S y_T$. Formally, we have a bijective map between our clause $U \in \mathcal{H}$ and the partition (S,T), and using the transformation $y_S = \prod_{i\in S} x_i$, for all x, there exists y such that $\phi(y) = \psi(x)$.

▶ **Definition 4.9.** *Let $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a semirandom k-XOR instance with constraint set $\mathcal{H} \subseteq [n]^k$. We call a 2-XOR instance ϕ an* f-induced 2-XOR instance *of ψ if each of its variables y_S is indexed by some $S \subseteq [n]^{\leqslant k}$ and there exists a set $\mathcal{B} = \mathcal{B}(\phi)$ of pairs of variables (S,T) and a bijection $f : \mathcal{B} \to \mathcal{H}$ such that ST is a permutation of $f(S,T)$ and ϕ can be written as*

$$\phi = \sum_{(S,T)\in\mathcal{B}} b_{f(S,T)} y_S y_T\,,$$

Let $d^{\phi}(S) = |\{U \in \mathcal{H} : f^{-1}(U) \ni S\}|$ denote the number of times the variable y_S appears in the above expression for ϕ, which we call the degree *of S in ϕ. Then we define the weighted adjacency matrix $A'(\phi)$ of ϕ with rows and columns indexed by variables of ϕ:*

$$(A'(\phi))_{S,T} = b_{f(S,T)}\mathbf{1}\{(S,T) \in \mathcal{B}\} + b_{f(T,S)}\mathbf{1}\{(T,S) \in \mathcal{B}\}\,.$$

Let $A(\phi)$ be a principal submatrix of $A'(\phi)$ and let $D(\phi)$ be the degree matrix of $A(\phi)$. We call $A(\phi)$ an induced (γ,δ)-spectral certificate for ψ *if the following hold:*

- *$A(\phi)$ can be formed by deleting rows of $A'(\phi)$ and their corresponding columns whose ℓ_1 norms sum to at most $\gamma \sum_{S,T} |A'(\phi)_{S,T}|$,*
- *$\rho(D^{-1}(\phi)A(\phi)) \leqslant n^{-\delta}$.*

A Sherali-Adams refutation of an induced 2-XOR instance of ψ implies a refutation of ψ itself, which we prove in the appendix.

▶ **Lemma 4.10.** *Let $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a k-XOR instance with constraint set $\mathcal{H} \subseteq [n]^k$ and ϕ be an f-induced 2-XOR instance of ψ. Suppose that for all R-local pseudodistributions $\mathcal{D}'$, we have $|\widetilde{\mathbb{E}}_{\mathcal{D}'}\phi(y)| \leqslant 2\gamma|\mathcal{H}|$. Then for all kR-local pseudodistributions $\mathcal{D}$, we have $|\widetilde{\mathbb{E}}_{\mathcal{D}}\psi(x)| \leqslant 2\gamma|\mathcal{H}|$.*

Using this, we prove that Sherali-Adams can refute XOR instances with an induced spectral certificate.

▶ **Lemma 4.11.** *Let $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a k-XOR instance and suppose there exists an induced (γ,δ)-spectral certificate for ψ. Then $R = O_{\gamma,\delta,k}(1)$-rounds of Sherali-Adams can certify that $\psi(x) \leqslant 2\gamma|\mathcal{H}|$; in other words, for all R-local pseudodistributions $\mathcal{D}$, $|\widetilde{\mathbb{E}}_{\mathcal{D}}\psi(x)| \leqslant 2\gamma|\mathcal{H}|$.*

Proof. Let ϕ be an f-induced 2-XOR instance of ψ and $A(\phi)$ be a corresponding induced (γ,δ)-spectral certificate. Let $R \coloneqq k \cdot R(A(\phi))$, where $R(A(\phi))$ is the quantity from Lemma 3.8 where $\eta = \gamma$, and let $\mathcal{D}$ be any $R = O_{\gamma,\delta,k}(1)$-local pseudodistribution. Since $\rho(D^{-1}(\phi)A(\phi)) \leqslant n^{-\delta}$, by Lemma 3.8, for any $R(A(\phi))$-local pseudodistribution $\mathcal{D}'$, we have

$$\begin{aligned}
|\widetilde{\mathbb{E}}_{\mathcal{D}'}\phi(y)| &= \frac{1}{2}\left|\widetilde{\mathbb{E}}y^\top A'(\phi)y\right| \\
&= \frac{1}{2}\left|\widetilde{\mathbb{E}}y^\top A(\phi)y\right| + \frac{1}{2}\left|\widetilde{\mathbb{E}}y^T(A'(\phi)-A(\phi))y\right| \\
&\leqslant \gamma|\mathcal{H}| + \gamma|\mathcal{H}| = 2\gamma|\mathcal{H}|\,,
\end{aligned}$$

so $\widetilde{\mathbb{E}}_{\mathcal{D}}\psi(x) \leqslant 2\gamma|\mathcal{H}|$ by Lemma 4.10. ◀

Furthermore, induced 2-XOR instances without too many variables admit an induced spectral certificate for its parent instance, which we show in Section 4.3.

▶ **Lemma 4.12.** *Let $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a semirandom k-XOR instance with constraint set $\mathcal{H} \subseteq [n]^k$ and ϕ be an f-induced 2-XOR instance of ψ with variable set $\mathcal{S}$ such that $|\mathcal{S}| \leqslant n^{-\delta}|\mathcal{H}|$. If $\gamma \geqslant n^{-\delta/2}$ and $4\sqrt{k\log n} \leqslant n^{\delta/20}$, then with high probability, there exists an induced $(\gamma, \frac{\delta}{5})$-spectral certificate for ψ with respect to $(\mathcal{B}, f)$.*

4.2 Proof of Theorem 4.2

Here we prove the main result of this section.

Proof of Theorem 4.2. First, suppose that k is even. For all $U = ST \in \mathcal{H}$, where $S, T \in [n]^{k/2}$ define $f(S,T) := U$. Construct the f-induced 2-XOR instance ϕ of ψ. Then we have at most $n^{k/2} = n^{-\delta}|\mathcal{H}|$ total variables, so by Lemma 4.12, there exists an induced $(\gamma, \frac{\delta}{5})$-spectral certificate for ψ with respect to f. Thus by Lemma 4.11, there exists some $R = O_{\gamma,\delta,k}(1)$ such that for all R-local pseudodistributions $\mathcal{D}$, we have $|\widetilde{\mathbb{E}}_{\mathcal{D}}\psi(x)| \leqslant 2\gamma|\mathcal{H}|$.

Now suppose that k is odd. We construct a partition $\mathcal{H} = \mathcal{H}' \sqcup \mathcal{H}'' \sqcup \mathcal{H}'''$ as follows:

1. While there exists some $S \in [n]^{(k-1)/2}$ and $i \in [n]$ such that there are more than $n^{\delta/2}$ total clauses of the form STi or TSi in $\mathcal{H}$, move all STi and TSi to a new hypergraph $\mathcal{H}'$.
2. Afterwards, while there exist some $(S,T) \in [n]^{(k-1)/2} \times [n]^{(k-1)/2}$ such that there are more than $n^{\delta/2}$ total clauses of the form $STi \in \mathcal{H}$, move all such STi from $\mathcal{H}$ to a new hypergraph $\mathcal{H}''$.
3. Let $\mathcal{H}'''$ consist of the remaining clauses.

Now, consider $\mathcal{H}'$. If $|\mathcal{H}'| \leqslant \frac{\gamma}{2}|\mathcal{H}|$, then $\psi(\mathcal{H}') \leqslant \frac{\gamma}{2}|\mathcal{H}|$ trivially. Otherwise, each edge $STi \in \mathcal{H}'$ consists of one part $(S,i) \in [n]^{(k-1)/2} \times [n]$ and one part $T \in [n]^{(k-1)/2}$. By construction, we have at most $n^{k/2+\delta/2}$ different (S,i) that occur in $\mathcal{H}'$, since each must occur in at least $n^{\delta/2}$ tuples for it to be added to $\mathcal{H}'$, and there are at most $n^{(k-1)/2}$ total T. Thus, when we use these clauses to construct an induced 2-XOR instance (letting $f((S,i),T) = STi$ if (S,i) was the heavy part and $f(S,(T,i)) = STi$ if (T,i) was the heavy part), we will have $\leqslant n^{k/2+\delta/2} + n^{(k-1)/2} \leqslant n^{-\delta/3}|\mathcal{H}'|$ total variables. Thus by Lemma 4.12, there exists an induced $(\frac{\gamma}{2}, \frac{\delta}{15})$-spectral certificate for $\psi(\mathcal{H}')$, so by Lemma 4.11 there exists some $R' = O_{\gamma/2,\delta/15,k}(1) = O_{\gamma,\delta,k}(1)$ such that for all R'-local pseudodistributions $\mathcal{D}'$, we have $\widetilde{\mathbb{E}}_{\mathcal{D}'}\psi(\mathcal{H}') \leqslant \gamma|\mathcal{H}'|$.

Similarly for $\mathcal{H}''$ either $|\mathcal{H}''| \leqslant \frac{\gamma}{2}|\mathcal{H}|$ and thus $\psi(\mathcal{H}'') \leqslant \frac{\gamma}{2}|\mathcal{H}|$, or we can construct an induced 2-XOR instance whose variables are $ST \in [n]^{k-1}$ which occur together in at least $n^{\delta/2}$ clauses and $i \in [n]$. Therefore we will have $\leqslant n^{k/2+\delta/2} + n \leqslant n^{-\delta/3}|\mathcal{H}''|$ total variables, so like before there exists an induced $(\frac{\gamma}{2}, \frac{\delta}{15})$-spectral certificate for $\psi(\mathcal{H}'')$ so for all R'-local pseudodistributions $\mathcal{D}'$, we have $\widetilde{\mathbb{E}}_{\mathcal{D}'}\psi(\mathcal{H}'') \leqslant \gamma|\mathcal{H}''|$.

Finally, for $\mathcal{H}'''$, if $|\mathcal{H}'''| \leqslant \frac{\gamma}{2}|\mathcal{H}|$, then $\psi(\mathcal{H}''') \leqslant \frac{\gamma}{2}|\mathcal{H}|$. Otherwise, construct a regular partition P of $\mathcal{H}'''$ by taking each $\mathcal{H}'''_i$ of size larger than $\frac{|\mathcal{H}'''|}{n}$ and repeatedly peeling off $\frac{|\mathcal{H}'''|}{n}$ at a time. Then $\mathcal{H}'''$ satisfies the conditions of Lemma 4.8 so there exists a $(\frac{\gamma}{2}, \frac{\delta}{9})$-spectral certificate for $\psi(\mathcal{H}''')$. Thus by Lemma 4.7, there exists some $R''' = O_{\gamma/2,\delta/9,k}(1) = O_{\gamma,\delta,k}(1)$ such that for all R'''-local pseudodistributions $\mathcal{D}'$, we have $\widetilde{\mathbb{E}}_{\mathcal{D}'}\psi(\mathcal{H}''') \leqslant \gamma|\mathcal{H}'''|$.

Combining the above and letting $R := \max(R', R''')$, for all R-local pseudodistributions $\mathcal{D}$, we have

$$\begin{aligned}\left|\widetilde{\mathbb{E}}_{\mathcal{D}}\psi(x)\right| &\leqslant \left|\widetilde{\mathbb{E}}_{\mathcal{D}}\psi(\mathcal{H}')(x)\right| + \left|\widetilde{\mathbb{E}}_{\mathcal{D}}\psi(\mathcal{H}'')(x)\right| + \left|\widetilde{\mathbb{E}}_{\mathcal{D}}\psi(\mathcal{H}''')(x)\right| \\ &\leqslant \frac{\gamma}{2}|\mathcal{H}| + \frac{\gamma}{2}|\mathcal{H}| + \gamma(|\mathcal{H}'| + |\mathcal{H}''| + |\mathcal{H}'''|) \\ &\leqslant 2\gamma|\mathcal{H}|\,.\end{aligned}$$

◀

4.3 Existence of spectral certificates

Here we prove Lemma 4.8 and Lemma 4.12. We use the following result, which we defer to the appendix, in the proof of Lemma 4.8.

▶ **Lemma 4.13.** *Let $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a semirandom k-XOR instance with constraint set $\mathcal{H} \subseteq [n]^k$ such that $|\mathcal{H}| = n^{k/2+\delta}$ and let P be a regular partition of the $\mathcal{H}_i$s. Suppose $\mathcal{H}$ has the following properties:*

- *For all $S \in [n]^{(k-1)/2}$, $i \in [n]$, there are at most $n^{\delta/2}$ total clauses of the form STi or TSi in $\mathcal{H}$.*
- *For all $S_1T_1 \in [n]^{(k-1)/2} \times [n]^{(k-1)/2}$, there are at most $n^{\delta/2}$ total clauses of the form $S_1T_1i \in \mathcal{H}$.*

Then with high probability, there exists a principal submatrix $A(\psi)$ of $A'(\psi)$ such that all of the following hold:

- *The deleted rows and columns of $A'(\psi)$ have ℓ_1 norms summing to at most*

$$30n^{-\delta/2} \sum_{i\in[n]} \sum_{j\in[p_i]} |\mathcal{H}_i^{(j)}|(|\mathcal{H}_i^{(j)}| - 1)\,.$$

- *Each row of $A(\psi)$ has at least n^δ nonzero entries.*
- *Each row of $A(\psi)$ has at least $\frac{1}{3}$ as many nonzero entries as the corresponding row of $A(\psi^{\mathbf{1}})$, where we recall that $A'(\psi^{\mathbf{1}})$ is the matrix defined with the same $\mathcal{H}$ as ψ but with all b_Us equal to 1.*
- *Each entry of $A(\psi)$ has absolute value at most $4kn^{\delta/4}\sqrt{\log n}$.*

▶ **Lemma 4.8.** *Let $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a semirandom k-XOR instance with constraint set $\mathcal{H} \subseteq [n]^k$ such that $|\mathcal{H}| = n^{k/2+\delta}$ and let P be a regular partition of the $\mathcal{H}_i$s. Suppose $\mathcal{H}$ has the following properties:*

- *For all $S \in [n]^{(k-1)/2}$, $i \in [n]$, there are at most $n^{\delta/2}$ total clauses of the form STi or TSi in $\mathcal{H}$.*
- *For all $S_1T_1 \in [n]^{(k-1)/2} \times [n]^{(k-1)/2}$, there are at most $n^{\delta/2}$ total clauses of the form $S_1T_1i \in \mathcal{H}$.*

Suppose that $\gamma \geqslant n^{-\delta/4}\sqrt{60}$ and $16384k^4 \log^3 n \leqslant n^{\delta/18}$. Then with high probability, there exists a $(\gamma, \delta/9)$-spectral certificate of ψ with respect to P.

Proof. Let $A(\psi)$ be the principal submatrix of $A'(\psi)$ given by Lemma 4.13. Since $\gamma \geqslant \sqrt{60}n^{-\delta/4}$, it satisfies the first condition of a (γ, δ)-spectral certificate with high probability. Now we prove the second condition. Let $D(\psi)$ be the degree matrix of $A(\psi)$, $D(\psi^{\mathbf{1}})$ be the degree matrix of $A(\psi^{\mathbf{1}})$, and $M(\psi)$ be the degree matrix of the support matrix of $A(\psi^{\mathbf{1}})$, i.e. a diagonal entry of $M(\psi)$ is the number of nonzero entries in the corresponding row of $A(\psi^{\mathbf{1}})$.

Note that we have $M(\psi) \leqslant 3D(\psi)$ entrywise. Define $\Gamma \coloneqq M(\psi)^{-1/2}A(\psi)M(\psi)^{-1/2}$, so we have

$$\rho(D(\psi)^{-1}A(\psi)) \leqslant 3\rho(M(\psi)^{-1}A(\psi)) = 3\rho(\Gamma) \leqslant 3\,\|\Gamma\|\,.$$

We start by upper bounding $\mathbb{E}\mathrm{tr}(\Gamma^\ell) = \mathbb{E}\mathrm{tr}((M^{-1}(\psi)A(\psi))^\ell)$ for even ℓ by analyzing closed length-ℓ walks $V_1 \mapsto V_2 \mapsto \cdots \mapsto V_\ell \mapsto V_1$ in the weighted multigraph whose vertices are variables of ψ corresponding to the weighted signed adjacency matrix $M^{-1}(\psi)A(\psi)$. Note that for any edge between two vertices $V_j = (S_1, S_2)$ and $V_{j+1} = (T_1, T_2)$, there exists some i such that either both $S_1T_1i, S_2T_2i \in \mathcal{H}$ or both $T_1S_1i, T_2S_2i \in \mathcal{H}$. Since each $U \in \mathcal{H}$ comes with an independent Rademacher sign ξ_U, if some clause $U \in \mathcal{H}$ appears only once as one of the two clauses inducing an edge in our walk, then the expected value of the walk is 0. Therefore, in a walk with a nonzero contribution to the expected trace, each clause which induces an edge must occur at least twice.

Then we can define an encoding of any closed walk $V_1 \mapsto V_2 \mapsto \cdots \mapsto V_\ell \mapsto V_1$ by first choosing a variable $V_j = (S_1, S_2)$, of which there are at most n^{k-1} choices, then choosing $b \in \{0,1\}^\ell$ such that b_j is the indicator that both of the two tuples which induce the edge from V_j to V_{j+1} have not induced a previous step in the walk. Since each edge must occur at least twice, we note that $b_j = 0$ for at least $\frac{\ell}{2}$ values of j.

At each step j, we choose the next step of the walk based on the value of b_j: At $V_j = (S_1, S_2)$, if $b_j = 1$, we simply pick an arbitrary neighbor of V_j, while if $b_j = 0$, there are at most $2j \leqslant 2\ell$ possible $U \in \mathcal{H}$ to choose from which have induced a previous step in the walk. Therefore, when $b_j = 1$, there are at most $D(\psi)(V_j)$ possible choices, so $\sum_{V_{j+1} \in N_1(V_j)} \frac{1}{M(\psi)(V_j)} \leqslant \frac{D(\psi)(V_j)}{M(\psi)(V_j)} \leqslant 4kn^{\delta/4}\sqrt{\log n}$. When $b_j = 0$, without loss of generality, suppose the tuple we pick is of the form $S_1 T_1 i \in \mathcal{H}$ for some $i \in [n]$. Then the second tuple inducing this edge must be of the form $(S_2, T_2, i) \in \mathcal{H}_i^{(j)}$ for some $i \in [n], j \in [p_i]$, and there are at most $n^{\delta/2}$ total choices for that, so $\sum_{V_{j+1} \in N_0(V_j)} \frac{1}{M(\psi)(V_j)} \leqslant \frac{2\ell n^{\delta/2}}{n^\delta} \leqslant 2\ell n^{-\delta/2}$, since $M(\psi)(V_j) \geqslant n^\delta$. Thus we get

$$\begin{aligned}
\mathbb{E}\mathrm{tr}(\Gamma^\ell) &\leqslant \sum_{b \in \{0,1\}^\ell} \sum_{V_1 \in [n]^{k-1}} \frac{1}{M(\psi)(V_1)} \sum_{S_2 \in N_{b_1}(S_1)} \frac{1}{M(\psi)(S_2)} \\
&\qquad \cdots \sum_{S_\ell \in N_{b_{\ell-1}}(S_{\ell-1})} \frac{\mathbf{1}\{S_1 \in N_{b_\ell}(S_\ell)\}}{M(\psi)(S_\ell)} \\
&\leqslant 2^\ell n^{k-1} (4kn^{\delta/4}\sqrt{\log n} \cdot 2\ell n^{-\delta/2})^{\ell/2} \\
&= 2^\ell n^{k-1} (2\sqrt{2k\ell}(\log n)^{1/4} n^{-\delta/8})^\ell \,.
\end{aligned}$$

Therefore, we set $\ell = 2\left\lceil \frac{k \log n}{2} \right\rceil$ so by Markov's inequality, we have

$$\Pr\left[\|\Gamma\| \geqslant n^{-\delta/9} \geqslant 8\sqrt{2}kn^{-\delta/8}\log^{3/4} n\right] \leqslant \frac{\mathbb{E}\mathrm{tr}(\Gamma^\ell)}{(8\sqrt{2}kn^{-\delta/8}\log^{3/4} n)^\ell} = \frac{1}{\mathrm{poly}(n^k)} \,. \qquad \blacktriangleleft$$

▶ **Lemma 4.12.** *Let $\psi(x) = \sum_{U \in \mathcal{H}} b_U \cdot x^U$ be a semirandom k-XOR instance with constraint set $\mathcal{H} \subseteq [n]^k$ and ϕ be an f-induced 2-XOR instance of ψ with variable set $\mathcal{S}$ such that $|\mathcal{S}| \leqslant n^{-\delta}|\mathcal{H}|$. If $\gamma \geqslant n^{-\delta/2}$ and $4\sqrt{k \log n} \leqslant n^{\delta/20}$, then with high probability, there exists an induced $(\gamma, \frac{\delta}{5})$-spectral certificate for ψ with respect to $(\mathcal{B}, f)$.*

Proof. While $A'(\phi)$ contains a row with at most $n^{\delta/2}$ entries, delete that row and its corresponding column, and let $A(\phi)$ be the result. Then note that the deleted rows contain at most $|\mathcal{S}|n^{\delta/2} \leqslant n^{-\delta/2}|\mathcal{H}|$ entries. Now let $D(\phi)$ be the degree matrix of $A(\phi)$, and note that all diagonal entries of $D(\phi)$ are at least $n^{\delta/2}$. Define $\Gamma = D(\phi)^{-1/2} A(\phi) D(\phi)^{-1/2}$, and note that $\rho(D(\phi)^{-1}A(\phi)) = \rho(\Gamma) \leqslant \|\Gamma\|$. We start by upper bounding $\mathbb{E}\mathrm{tr}(\Gamma^\ell) = \mathbb{E}\mathrm{tr}((D^{-1}(\phi)A(\phi))^\ell)$ by analyzing closed even length-ℓ walks $S_1 \mapsto S_2 \mapsto \cdots \mapsto S_\ell \mapsto S_1$ in the weighted graph whose vertices are variables of ϕ which corresponds to the weighted signed adjacency matrix $(D^{-1}(\phi)A(\phi)$. Note that there exists an edge between two vertices S_i, S_{i+1} if either $f(S_i, S_{i+1}) \in \mathcal{H}$ or $f(S_{i+1}, S_i) \in \mathcal{H}$. Since each $U \in \mathcal{H}$ comes with an independent Rademacher sign ξ_U, if some unordered tuple (S, T) appears only once as an edge of the walk, then the expected value of the walk is 0. Therefore, each edge in a walk with a nonzero contribution to the expected trace must occur at least twice. Then we can define an encoding of any closed walk $S_1 \mapsto S_2 \mapsto \cdots \mapsto S_\ell \mapsto S_1$ by first choosing a variable S_1, of which there are $|[n]^{k-1}| < n^k$ choices, then choosing $b \in \{0,1\}^\ell$ such that b_i is the indicator that the ith edge of the walk has not occurred in a previous step. Since each edge must occur at least twice, we note that $b_i = 0$ for at least $\frac{\ell}{2}$ values of i.

At each step i, we choose the next step of the walk based on the value of b_i: At S_{i-1}, if $b_i = 1$, we simply pick an arbitrary neighbor of S_{i-1}, while if $b_i = 0$, we pick some S_i such that the edge (S_{i-1}, S_i) has occurred previously. Then, when $b_i = 1$, there are at most $D(\phi)(S_{i-1})$ possible choices, so $\sum_{S_i \in N_1(S_{i-1})} 1/D(\phi)(S_{i-1}) \leqslant 1$. When $b_i = 0$, there are at most $i - 1 \leqslant \ell$ previous edges to choose from, so $\sum_{S_i \in N_0(S_{i-1})} 1/D(\phi)(S_{i-1}) \leqslant \ell n^{-\delta/2}$. Combining the above, we get

$$\begin{aligned}\mathbb{E}\mathrm{tr}(\Gamma^\ell) &\leqslant \sum_{b \in \{0,1\}^\ell} \sum_{S_1 \in \mathcal{S}} \frac{1}{D(\phi)(S_1)} \sum_{S_2 \in N_{b_1}(S_1)} \frac{1}{D(\phi)(S_2)} \cdots \sum_{S_\ell \in N_{b_{\ell-1}}(S_{\ell-1})} \frac{\mathbf{1}\{S_1 \in N_{b_\ell}(S_\ell)\}}{D(\phi)(S_\ell)} \\ &\leqslant 2^\ell n^k (\ell n^{-\delta/2})^{\ell/2} \,.\end{aligned}$$

Therefore, we set $\ell = 2\left\lceil \frac{3k \log_2 n}{4} \right\rceil$ so by Markov's inequality, we have

$$\Pr\left[\|\Gamma\| \geqslant n^{-\delta/5} \geqslant 4n^{-\delta/4}\sqrt{k \log_2 n}\right] \leqslant \frac{\mathbb{E}\mathrm{tr}(\Gamma^\ell)}{(4n^{-\delta/4}\sqrt{k \log_2 n})^\ell} = \frac{1}{\mathrm{poly}(n^k)} \,. \qquad \blacktriangleleft$$

References

1. Jackson Abascal, Venkatesan Guruswami, and Pravesh K. Kothari. Strongly refuting all semi-random boolean CSPs. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA 2021)*, pages 454–472. SIAM, 2021. doi:10.1137/1.9781611976465.28.
2. Sarah R. Allen, Ryan O'Donnell, and David Witmer. How to refute a random CSP. In *56th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2015)*, pages 689–708. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.48.
3. Boaz Barak and Ankur Moitra. Noisy tensor completion via the sum-of-squares hierarchy. In *Proceedings of the 29th Conference on Learning Theory (COLT 2016)*, volume 49 of *JMLR Workshop and Conference Proceedings*, pages 417–445. JMLR.org, 2016. URL: https://proceedings.mlr.press/v49/barak16.html.
4. Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Integrality gaps for Sherali-Adams relaxations. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC 2009)*, pages 283–292. ACM, 2009. doi:10.1145/1536414.1536455.
5. Amin Coja-Oghlan, Andreas Goerdt, and André Lanka. Strong refutation heuristics for random k-SAT. *Combinatorics, Probability and Computing*, 16(1):5–28, 2007. doi:10.1017/S096354830600784X.
6. Tommaso d'Orsi and Luca Trevisan. An Ihara-Bass formula for non-boolean matrices and strong refutations of random CSPs. In *38th Computational Complexity Conference (CCC 2023)*, volume 264 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 27:1–27:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.CCC.2023.27.
7. Noah Fleming, Pravesh Kothari, and Toniann Pitassi. Semialgebraic proofs and efficient algorithm design. *Foundations and Trends in Theoretical Computer Science*, 14(1–2):1–221, 2019. doi:10.1561/0400000086.
8. Dmitry Gavinsky, Shachar Lovett, Michael Saks, and Srikanth Srinivasan. A tail bound for read-k families of functions. *Random Structures and Algorithms*, 47(1):99–108, 2015. doi:10.1002/rsa.20532.
9. Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995. doi:10.1145/227683.227684.
10. Venkatesan Guruswami, Pravesh K. Kothari, and Peter Manohar. Algorithms and certificates for boolean CSP refutation: Smoothed is no harder than random. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC 2022)*, pages 678–689. ACM, 2022. doi:10.1145/3519935.3519955.

11 Samuel B. Hopkins, Tselil Schramm, and Luca Trevisan. Subexponential LPs approximate max-cut. In *61st Annual IEEE Symposium on Foundations of Computer Science (FOCS 2020)*, pages 943–953. IEEE, 2020. doi:10.1109/FOCS46700.2020.00092.

12 Jun-Ting Hsieh, Pravesh K. Kothari, and Sidhanth Mohanty. A simple and sharper proof of the hypergraph moore bound. In *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms (SODA 2023)*, pages 2324–2344. SIAM, 2023. doi:10.1137/1.9781611977554.CH89.

13 Ryan O'Donnell and Tselil Schramm. Sherali-Adams strikes back. In *34th Computational Complexity Conference (CCC 2019)*, volume 137 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 8:1–8:30. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.CCC.2019.8.

A Proofs of auxiliary statements

In this section, we fill in the proofs of Lemma 3.7 and Lemma 3.8. The proofs here are all reformulations of ideas in prior works [2, 13] and are included here for completeness.

A.1 Auxiliary statements for random k-XOR

We show that with high probability, the matrix $A(\psi)$ defined in Definition 3.6 is a spectral certificate.

▶ **Lemma 3.7.** *Suppose ψ is a random k-XOR instance with $n^{k/2+\delta}$ constraints such that $2^{O(k)}\log^3 n \leqslant n^{\delta/2}$. Then $A(\psi)$ is a $\frac{\delta}{2}$-spectral certificate for ψ with high probability.*

This follows directly from the following two statements.

▶ **Fact A.1** ([2, Lemma A.5])**.** *Let $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a k-XOR instance with $n^{k/2+\delta}$ expected constraints. Then with high probability, $\|A(\psi)\| \leqslant 2^{O(k)} n^{\delta}\log^3 n$.*

▶ **Proposition A.2.** *Let $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a k-XOR instance with $n^{k/2+\delta}$ expected constraints. Then with probability $1 - 1/\mathrm{poly}(n)$, we have $\min_{(S_1,S_2)} \left\|A(\psi)_{(S_1,S_2)}\right\|_1 \geqslant \min\left(\frac{n^{2\delta}}{4}, \frac{n}{6}\right)$.*

Proof. First, suppose that $\delta < \frac{1}{2}$. Note that each $U \in [n]^k$ is included in $\mathcal{H}$ independently with probability $p = n^{-k/2+\delta}$. For all $S \in [n]^{(k-1)/2}$ and $i \in [n]$, the number of $(STi) \in \mathcal{H}$ is Binomial$(n^{(k-1)/2}, n^{-k/2+\delta})$. Then this is nonzero with probability $1-(1-n^{-k/2+\delta})^{n^{(k-1)/2}} \geqslant 1 - e^{-n^{\delta-1/2}} \geqslant n^{\delta-1/2}(1-o(1))$. Recall the notation $\psi^{\mathbf{1}}$ from Definition 4.5, and note that for any $(S_1,S_2) \in [n]^{(k-1)/2}\times[n]^{(k-1)/2}$, $A(\psi^{\mathbf{1}})_{(S_1,S_2)}$ has a total of $\sum_{i\in[n]} \mathbf{1}\{\exists S_1 Ti \in \mathcal{H}\}\mathbf{1}\{\exists S_2 Ti\in\mathcal{H}\}$ nonzero entries, which is Binomial with mean $n^{2\delta}(1-o(1))$ when $S_1 \neq S_2$ or mean $n^{\frac{1}{2}+\delta}(1-o(1)) \geqslant n^{2\delta}$ when $S_1 = S_2$, and thus it has at least $n^{2\delta}/2$ nonzero entries by Chernoff with probability $1-\exp(-\Omega(n^{2\delta}))$. Recall that if an entry of $A(\psi^{\mathbf{1}})$ is nonzero, then the corresponding entry of $A(\psi)$ is nonzero with probability at least $\frac{1}{2}$ (see Remark 4.6), so $A(\psi)_{(S_1,S_2)}$ has at least $n^{2\delta}/4$ nonzero entries with probability $1-\exp(-\Omega(n^{2\delta}))$. The result thus holds simultaneously for all $(S_1,S_2)\in[n]^{(k-1)/2}\times[n]^{(k-1)/2}$ by a union bound.

Now suppose that $\delta \geqslant \frac{1}{2}$. Then following the $\delta < \frac{1}{2}$ case, the number of $(STi) \in \mathcal{H}$ is nonzero with probability at least $1-(1-n^{-k/2+\frac{1}{2}})^{n^{(k-1)/2}} \geqslant 1-\frac{1}{e}-o(1)$. Then $\sum_{i\in[n]}\mathbf{1}\{\exists S_1 Ti\in\mathcal{H}\}\mathbf{1}\{\exists S_2 Ti\in\mathcal{H}\}$ is Binomial with mean at least $(1-\frac{1}{e}-o(1))^2 n$, and thus $A(\psi^{\mathbf{1}})_{(S_1,S_2)}$ has at least $n/3$ nonzero entries with probability $1-\exp(-\Omega(n))$ by Chernoff, and thus $A(\psi)_{(S_1,S_2)}$ has at least $n/6$ nonzero entries with probability $1-\exp(-\Omega(n))$, and as before this holds simultaneously for all (S_1,S_2) by a union bound. ◀

▶ **Lemma 3.8** (Abstracting out [13]). *Let A be a symmetric matrix with integer entries with rows and columns indexed by t-tuples of $[n]$, and let D be its degree matrix. Let $\pi_* = \frac{1}{\sum_{S,T}|A_{S,T}|}\min_S \|A_S\|_1$, $\rho(\cdot)$ denote the spectral radius of a matrix, and $\eta > \min(\pi_*^{-1/2}, \rho(D^{-1}A))$. Take $\ell \coloneqq \left\lceil \frac{1}{4}\frac{\log(\eta^2\pi_*)}{\log(\rho(D^{-1}A)/\eta)} \right\rceil$ and $p \coloneqq \lceil (\frac{2}{5\eta})^{2\ell} \rceil$. Then for $R(A) \coloneqq tp\ell + t$, any R-local pseudoexpectation $\mathcal{D}$ on $x_1, \cdots, x_n$ satisfies*

$$\left| \widetilde{\mathbb{E}}_{\mathcal{D}} (x^{\odot t})^\top A x^{\odot t} \right| \leqslant \eta \sum_{S,T} |A_{S,T}|,$$

where $x^{\odot t}$ is the n^t-dimensional vector whose entries are indexed by t-tuples of $[n]$ and are equal to the product of the corresponding entries.

We utilize the following reparameterization of results from [13]:

▶ **Fact A.3** ([13, Theorem 3.6, Corollary 3.7]). *Let $G = (V, E)$ be a multigraph with signs b_{uv} for all $uv \in E$ and signed transition operator $\overline{K}$. Suppose we have $\pi_* = \min_{v \in V} \frac{\deg(v)}{2|E|}$. Given $\gamma > \min(\pi_*^{-1/2}, \rho(\overline{K}))$, take $\ell = \left\lceil \frac{1}{4}\frac{\log(\gamma^2\pi_*)}{\log(\rho(\overline{K})/\gamma)} \right\rceil$, and take $k = \lceil (\frac{2}{5\gamma})^{2\ell} \rceil$. Then for $R = k\ell + 1$, it holds that R-local degree-2 SOS can deduce that for all $x \in \{\pm 1\}^V$,*

$$-\gamma \leqslant \frac{1}{|E|} \sum_{uv \in E} b_{uv} x_u x_v \leqslant \gamma\,.$$

Proof of Lemma 3.8. For all $S \subseteq [n]^t$, define a variable y_S. Define an R-local pseudoexpectation $\mathcal{D}'$ over the y_Ss by setting, for $r \leqslant R$,

$$\widetilde{\mathbb{E}}_{\mathcal{D}'}[y_{S_1} y_{S_2} \cdots y_{S_r}] \coloneqq \widetilde{\mathbb{E}}_{\mathcal{D}} \prod_{i \in S_1} x_i \prod_{i \in S_2} x_i \cdots \prod_{i \in S_r} x_i\,.$$

Applying Fact A.3 with $\eta = \gamma$, we see that $\widetilde{\mathbb{E}}_{\mathcal{D}'} \sum_{S,T} y_S y_T A_{S,T} \leqslant \eta \sum_{S,T} |A_{S,T}|$, and therefore

$$\begin{aligned}\widetilde{\mathbb{E}}_{\mathcal{D}} (x^{\odot t})^\top A x^{\odot t} &= \sum_{S,T} A_{S,T} \widetilde{\mathbb{E}}_{\mathcal{D}} \prod_{i \in S} x_i \prod_{i \in T} x_i \\ &= \widetilde{\mathbb{E}}_{\mathcal{D}'} \sum_{S,T} A_{S,T} y_S y_T \leqslant \eta \sum_{S,T} |A_{S,T}|\,.\end{aligned}$$ ◀

A.2 Auxiliary statements for semirandom k-XOR

▶ **Lemma 4.10.** *Let $\psi(x) = \sum_{U \in \mathcal{H}} b_U \cdot x^U$ be a k-XOR instance with constraint set $\mathcal{H} \subseteq [n]^k$ and ϕ be an f-induced 2-XOR instance of ψ. Suppose that for all R-local pseudodistributions $\mathcal{D}'$, we have $|\widetilde{\mathbb{E}}_{\mathcal{D}'} \phi(y)| \leqslant 2\gamma|\mathcal{H}|$. Then for all kR-local pseudodistributions $\mathcal{D}$, we have $|\widetilde{\mathbb{E}}_{\mathcal{D}} \psi(x)| \leqslant 2\gamma|\mathcal{H}|$.*

Proof. Let $\mathcal{D}$ be a kR-local pseudodistribution over variables $x_1, \cdots, x_n$. Letting $\mathcal{S}$ be the variable set of ϕ, define an R-local pseudodistribution $\mathcal{D}'$ over variables y_S of ϕ by setting, for $r \leqslant R$,

$$\widetilde{\mathbb{E}}_{\mathcal{D}'}[y_{S_1} y_{S_2} \cdots y_{S_r}] \coloneqq \widetilde{\mathbb{E}}_{\mathcal{D}} \prod_{i \in S_1} x_i \prod_{i \in S_2} x_i \cdots \prod_{i \in S_r} x_i\,.$$

Then we have

$$\widetilde{\mathbb{E}}_{\mathcal{D}} \psi(x) = \widetilde{\mathbb{E}}_{\mathcal{D}} \sum_{(S,T) \in \mathcal{B}} b_{f(S,T)} \prod_{i \in S} x_i \prod_{i \in T} x_i = \widetilde{\mathbb{E}}_{\mathcal{D}'} \sum_{(S,T) \in \mathcal{B}} b_{f(S,T)} y_S y_T = \widetilde{\mathbb{E}}_{\mathcal{D}'} \phi(y) \leqslant 2\gamma|\mathcal{H}|\,.$$ ◀

▶ **Lemma 4.13.** *Let $\psi(x) = \sum_{U\in\mathcal{H}} b_U \cdot x^U$ be a semirandom k-XOR instance with constraint set $\mathcal{H} \subseteq [n]^k$ such that $|\mathcal{H}| = n^{k/2+\delta}$ and let P be a regular partition of the $\mathcal{H}_i$s. Suppose $\mathcal{H}$ has the following properties:*

- *For all $S \in [n]^{(k-1)/2}$, $i \in [n]$, there are at most $n^{\delta/2}$ total clauses of the form STi or TSi in $\mathcal{H}$.*
- *For all $S_1T_1 \in [n]^{(k-1)/2} \times [n]^{(k-1)/2}$, there are at most $n^{\delta/2}$ total clauses of the form $S_1T_1i \in \mathcal{H}$.*

Then with high probability, there exists a principal submatrix $A(\psi)$ of $A'(\psi)$ such that all of the following hold:

- *The deleted rows and columns of $A'(\psi)$ have ℓ_1 norms summing to at most*

$$30n^{-\delta/2} \sum_{i\in[n]} \sum_{j\in[p_i]} |\mathcal{H}_i^{(j)}|(|\mathcal{H}_i^{(j)}| - 1)\,.$$

- *Each row of $A(\psi)$ has at least n^δ nonzero entries.*
- *Each row of $A(\psi)$ has at least $\frac{1}{3}$ as many nonzero entries as the corresponding row of $A(\psi^{\mathbf{1}})$, where we recall that $A'(\psi^{\mathbf{1}})$ is the matrix defined with the same $\mathcal{H}$ as ψ but with all b_Us equal to 1.*
- *Each entry of $A(\psi)$ has absolute value at most $4kn^{\delta/4}\sqrt{\log n}$.*

We use the following concentration result for dependent indicators.

▶ **Fact A.4** (Chernoff Bound for Weakly Dependent Random Variables). *[8] Suppose we have a set S of independent random variables and indicator random variables $Y_1, \cdots, Y_r$ with mean μ, such that each Y_i is a function of some $T_i \subseteq S$ and no $s \in S$ is contained in more than k total T_i's. Then for any $\varepsilon > 0$, we have*

$$\Pr[Y_1 + \cdots + Y_r \notin (\mu \pm \varepsilon)r] \leqslant 2e^{-2\varepsilon^2 r/k}\,.$$

Proof of Lemma 4.13. Note that since P has at most $2n$ parts, the total ℓ_1 norm of $A'(\psi^{\mathbf{1}})$ is $\sum_i \sum_{j\in[p_i]} \binom{|\mathcal{H}_i^{(j)}|}{2} \geqslant 2n \cdot \frac{1}{2}\left(\frac{|\mathcal{H}|}{2n}\right)^2 (1-o(1)) \geqslant \frac{1}{5}n^{k-1+2\delta}$. Also, since for all $(S_1,S_2),(T_1,T_2)$, there are at most $n^{\delta/2}$ total i such that $S_1T_1i \in \mathcal{H}$ or $S_2T_2i \in \mathcal{H}$, there are at most $n^{\delta/2}$ pairs of clauses that contribute to each entry of $A'(\psi)$; in other words, every entry of $A'(\psi^{\mathbf{1}})$ is at most $n^{\delta/2}$. Consider an arbitrary entry $(S_1,S_2),(T_1,T_2)$. We note that $A(\psi)_{(S_1,S_2),(T_1,T_2)}$ is the sum of $A(\psi^{\mathbf{1}})_{(S_1,S_2),(T_1,T_2)}$ products of independent Rademacher random variables, all of which are distinct, and thus it is also the sum of independent Rademachers, so $A(\psi)_{(S_1,S_2),(T_1,T_2)} \leqslant 4k\sqrt{A(\psi^{\mathbf{1}})_{(S_1,S_2),(T_1,T_2)}\log n} \leqslant 4kn^{\delta/4}\sqrt{\log n}$ with probability $1 - \frac{1}{n^{2k}}$ and thus it holds for all n^{2k-2} entries with probability $\geqslant 1 - 1/n^2$.

Now, while there exists some row of $A'(\psi^{\mathbf{1}})$ indexed by (S_1,S_2) whose total sum is less than $3n^{3\delta/2}$, delete that row and its corresponding column. Then the sum of the elements of $A'(\psi^{\mathbf{1}})$ we delete is at most $2 \cdot 3n^{k-1+3\delta/2} \leqslant 30n^{-\delta/2}\sum_{i\in[n]}\sum_{j\in[p_i]}|\mathcal{H}_i^{(j)}|(|\mathcal{H}_i^{(j)}|-1)$, and first condition then follows from the fact that the ℓ_1 norm of rows in $A'(\psi)$ is upper bounded by the sum of the corresponding row in $A'(\psi^{\mathbf{1}})$. Let $A(\psi)$ and $A(\psi^{\mathbf{1}})$ be the matrices after the deletion step.

Now consider a surviving row indexed by some (S_1,S_2), whose ℓ_1 norm is at least $3n^{3\delta/2}$. Since every entry of $A(\psi^{\mathbf{1}})$ is at most $n^{\delta/2}$, there are at least $3n^\delta$ nonzero entries in the row indexed by (S_1,S_2), each of which are indexed by some (T_1,T_2). Recall that if an entry $A(\psi^{\mathbf{1}})_{(S_1,S_2),(T_1,T_2)}$ is nonzero, then $A(\psi)_{(S_1,S_2),(T_1,T_2)}$ is nonzero with probability at least $\frac{1}{2}$ (Remark 4.6). Furthermore, each entry in row (S_1,S_2) of $A(\psi)$ depends on variables of the form b_{S_1Ti} and b_{S_2Ti}. Note that anytime some b_{S_1Ti} appears in row (S_1,S_2), it must be multiplied with some $b_{S_2T'i}$ (in which case it corresponds to the (T,T') entry of that

row). However, we are given that there are at most $n^{\delta/2}$ total $S_2T'i \in \mathcal{H}$, and thus b_{S_1Ti} appears in at most $n^{\delta/2}$ entries in row (S_1, S_2). Therefore, the family of indicators of entries in row (S_1, S_2) being nonzero is read-$n^{\delta/2}$ with respect to the b_Us, and thus after coupling to a family of indicators which occur with probability $\frac{1}{2}$, by Fact A.4, with probability $1 - \exp(-\Omega(n^{\delta/2}))$, the number of nonzero entries in row (S_1, S_2) of $A(\psi)$ is at least $\frac{1}{3}$ the number of nonzero entries in row (S_1, S_2) of $A(\psi^{\mathbf{1}})$, which is at least n^{δ}. Union bounding over all $\leqslant n^{k-1}$ rows of $A'(\psi)$ gives the desired result. ◀

Local Algorithms and the Failure of Log-Depth Quantum Advantage on Sparse Random CSPs

Antares Chen
University of Chicago, IL, USA

Neng Huang
University of Michigan, Ann Arbor, MI, USA

Kunal Marwaha
University of Chicago, IL, USA

Abstract

We construct and analyze a message-passing algorithm for random constraint satisfaction problems (CSPs) at large clause density, generalizing work of El Alaoui, Montanari, and Sellke for Maximum Cut [3] through a connection between random CSPs and mean-field Ising spin glasses [1, 55]. For CSPs with even predicates, the algorithm asymptotically solves a stochastic optimal control problem dual to an extended Parisi variational principle. This gives an optimal fraction of satisfied constraints among algorithms obstructed by the branching overlap gap property of Huang and Sellke [50], notably including the Quantum Approximate Optimization Algorithm and all quantum circuits on a bounded-degree architecture of up to $\epsilon \cdot \log n$ depth [24].

2012 ACM Subject Classification Theory of computation → Randomness, geometry and discrete structures; Theory of computation → Design and analysis of algorithms

Keywords and phrases Random CSP, message-passing algorithm

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.78

Category RANDOM

Related Version *Full Version*: https://arxiv.org/abs/2310.01563

1 Introduction

Today's quantum computers are starting to show signs of advantage over classical machines. Many people are seeking to use this technology in a practical application as soon as possible. One such area is combinatorial optimization: even though the problems are NP-hard, optimization has incredible value to the contemporary world. This industry is also eager to experiment with quantum computers, perhaps because rigorous worst-case analysis is not always relevant.

The most popular quantum protocol for classical optimization is the *Quantum Approximate Optimization Algorithm* (QAOA), proposed in [34]. Given an optimization problem, the protocol associates a qubit to each variable, and evolves a quantum state to a near-ground state of the problem, depending on choice of hyperparameters. The QAOA has also sparked the interest of theorists, characterizing the protocol's performance on canonical problems, and debating whether it is any more powerful than existing classical algorithms (e.g. [9, 45, 11]).

One key property of the QAOA is its *locality*: at each layer of the algorithm, a qubit interacts only with qubits associated with adjacent variables in the optimization problem. Locality is known to limit the QAOA up to logarithmic circuit depth when the optimization problem is *sparse* [32, 33]. This limitation is due to a phenomenon known as the *overlap gap property* (OGP), a type of clustering behavior of near-optimal solutions [44]. Notably, this is an *average-case* property of many random optimization problems. Recent formulations of the OGP have led to quantitative obstructions for broad families of algorithms, both quantum and classical [21, 24, 50].


© Antares Chen, Neng Huang, and Kunal Marwaha;
licensed under Creative Commons License CC-BY 4.0

In this work, we show that *with high probability* over a large family of sparse optimization problems, there is an *optimal* algorithm among those obstructed by the OGP. Furthermore, this algorithm is *classical*, so there can be no quantum advantage by the QAOA up to logarithmic depth. In fact, this no-go result extends to logarithmic-depth quantum circuits on fixed, bounded-degree architectures, such as the *brickwork* circuits [18] used in random circuit sampling [15]. This is the strongest *average-case* result known to the authors regarding quantum advantage in classical optimization.

Our algorithm is related to the theory of *message-passing* used in random inference problems [13], and inspired by rich connections to statistical physics models such as the mean-field spin glass [82]. The parameters of the algorithm are chosen by solving a stochastic control problem dual to a problem determining the quantitative barrier of the OGP, as pioneered in [1, 50]. In particular, our algorithm extends work of [3], and uses a sparse-to-dense connection observed in e.g. [26, 55]. The performance of the algorithm, like that of [68, 1, 3, 79], is dependent on a technical condition (Assumption 1) which we explain in the text.

1.1 Background

We study Constraint Satisfaction Problems (CSPs). A CSP instance consists of n variables and a set of constraints associated with a *predicate* $f : \{\pm 1\}^r \to \{0, 1\}$.

▶ **Definition 1** (Instance of a random CSP)**.** *Consider a function $f : \{\pm 1\}^r \to \{0, 1\}$. Fix a constant $\alpha > 0$. A* random CSP instance $\mathcal{I}$ *over n variables $\{x_i\}_{i\in[n]}$ and $m = \alpha \cdot n$ clauses is generated by the following procedure: For each $i = 1, \dots, m$, draw $i_1, \dots, i_r$ uniformly i.i.d from $[n]$, draw r random signs $\epsilon_{i_1}, \dots, \epsilon_{i_r}$ uniformly i.i.d from $\{\pm 1\}$, and add the constraint e to $E(\mathcal{I})$, where e describes the clause $f_e(x_e) \stackrel{\text{def}}{=} f(\epsilon_{i_1} x_{i_1}, \dots, \epsilon_{i_r} x_{i_r})$. A clause is satisfied if it evaluates to* 1.

We denote this random model as $\mathsf{CSP}_f(\alpha)$. We are interested in the best *satisfying fraction* (i.e. largest fraction of clauses satisfied by some assignment in $\{\pm 1\}^n$) an algorithm can achieve as $n \to \infty$ when α is a very large constant, i.e. so that instances of $\mathsf{CSP}_f(\alpha)$ are typically unsatisfiable. This value is related to the rich literature of *mean-field Ising spin glasses*. We summarize some relevant results, especially drawing on [1, 3, 55]. Let $\mathscr{U}$ denote the following set of functions:

$$\mathscr{U} \stackrel{\text{def}}{=} \left\{ \mu : [0, 1) \to \mathbb{R}_{\geq 0} : \mu \text{ is non-decreasing, right-continuous, and } \int_0^1 \mu(t)\, dt < \infty \right\}.$$

Given a univariate polynomial $\xi : \mathbb{R} \to \mathbb{R}$ with non-negative coefficients, define the functional $\mathsf{P}_\xi : \mathscr{U} \to \mathbb{R}$ given by

$$\mathsf{P}_\xi(\mu) \stackrel{\text{def}}{=} \Phi^\mu(0, 0) - \frac{1}{2} \cdot \int_0^1 \xi''(t) \cdot t \cdot \mu(t)\, dt\,,$$

where $\Phi^\mu : [0, 1) \times \mathbb{R} \to \mathbb{R}$ is a solution of the following partial differential equation:[1]

$$\begin{aligned} &\Phi^\mu_t(t, x) = -\frac{\xi''(t)}{2} \cdot \left(\Phi^\mu_{xx}(t, x) + \mu(t) \cdot \left(\Phi^\mu_x(t, x) \right)^2 \right) && \forall t \in [0, 1), \forall x \in \mathbb{R}\,, \\ &\Phi^\mu(1, x) = |x| && \forall x \in \mathbb{R}\,. \end{aligned} \tag{1.1}$$

P_ξ is known as the Parisi functional, while Equation (1.1) are known as the Parisi equations [73]. Let the infimum of P_ξ be defined as $\mathsf{OPT}_\xi \stackrel{\text{def}}{=} \inf_{\mu\in\mathscr{U}} \mathsf{P}_\xi(\mu)$.

[1] For the multivariable function $\Phi^\mu(t, x)$, we use subscripts to denote partial derivatives.

▶ **Theorem 2** ([6, 53]). *Consider any polynomial ξ with non-negative coefficients. Then OPT_ξ is achieved by some unique $\mu \in \mathcal{U}$.*

Surprisingly, the maximum satisfying fraction of a typical instance $\mathcal{I} \sim \mathsf{CSP}_f(\alpha)$ is related to OPT_ξ:

▶ **Theorem 3** ([55, Corollary 1.2], building on [73, 85, 70, 71, 26, 80, 72]). *Consider $\mathcal{I} \sim \mathsf{CSP}_f(\alpha)$. With high probability as $n \to \infty$, the maximum satisfying fraction of $\mathcal{I}$ is*[2]

$$\mathbb{E}[f] + \frac{\mathsf{OPT}_\xi}{\sqrt{\alpha}} + o_\alpha\left(\frac{1}{\sqrt{\alpha}}\right),$$

where $\xi(s) \stackrel{\text{def}}{=} \sum_{j=1}^{r} \|f^{=j}\|^2 s^j$, and $\|f^{=j}\|^2$ is the Fourier weight of f at degree j.

We also know that some algorithms are quantitatively obstructed due to the geometry of the solution space.

▶ **Definition 4** (Overlap-concentrated algorithm, [50, Definition 2.1]). *A deterministic algorithm $\mathcal{A}$ is* overlap-concentrated *with respect to a random family of problems $\mathcal{F}$ if the following is true for every $t \in [0,1]$ and constant $\delta > 0$: Draw two instances $\mathcal{I}_1, \mathcal{I}_2 \sim \mathcal{F}$ that are t-correlated.*[3] *Let $\mathcal{A}(\mathcal{I})$ be the assignment in $\{\pm 1\}^n$ that $\mathcal{A}$ generates on instance $\mathcal{I}$. Then the event*

$$\frac{1}{n}\left|\langle \mathcal{A}(\mathcal{I}_1), \mathcal{A}(\mathcal{I}_2)\rangle - \mathop{\mathbb{E}}_{\mathcal{I}_1,\mathcal{I}_2}[\langle \mathcal{A}(\mathcal{I}_1), \mathcal{A}(\mathcal{I}_2)\rangle]\right| \le \delta \tag{1.2}$$

occurs with high probability over choice of $\mathcal{I}_1, \mathcal{I}_2$.

For example, *local* algorithms in the factors of i.i.d. model and certain quantum circuits up to logarithmic depth [55] are overlap-concentrated on $\mathsf{CSP}_f(\alpha)$. A randomized algorithm is overlap-concentrated if the w.h.p. statement and the expectation in Equation (1.2) also vary over the algorithm's internal randomness.

We briefly explain the intuition of why Definition 4 prevents algorithms from succeeding on certain optimization problems; see [40] for more detail. Suppose the optimal assignments are *clustered*; i.e. they are "needles in the haystack". Then random instances will have "needles" in random locations in the solution space. But for $t \in [0,1]$, consider instances $\mathcal{I}_2(t)$ that are t-correlated with the initial instance $\mathcal{I}_1$. If the algorithm $\mathcal{A}$ is successful, the correlation $\frac{1}{n}\mathbb{E}_{\mathcal{I}_2(t)}[\langle \mathcal{A}(\mathcal{I}_1), \mathcal{A}(\mathcal{I}_2(t))\rangle]$ measures the correlation of these "needles" (i.e. good assignments), and so it varies smoothly from 0 to 1 as t goes from 0 to 1. The *overlap-concentrated* property is a bound on the variance of this quantity, meaning that with high probability over $\{\mathcal{I}_2(t)\}_{t\in[0,1]}$, the quantity $\frac{1}{n}\langle \mathcal{A}(\mathcal{I}_1), \mathcal{A}(\mathcal{I}_2(t))\rangle$ varies smoothly with t from 0 to 1. But this is not possible on problems that exhibit an *overlap gap property* (OGP)[4] across pairs of instances; so $\mathcal{A}$ cannot be successful at locating the "needles" or good assignments.

There is an emerging literature generalizing the *overlap gap property* with increasingly strong quantitative obstructions [44, 21, 43, 50]. We use the *branching OGP* of [50], which gives the strongest quantitative bounds of algorithms known on spin glasses and random CSPs [55].

---

[2] Following [26], we use $o_x(g(x))$ to mean the value is at most some function $\kappa(n,x)$ satisfying $\frac{\kappa(n,x)}{g(x)} \to 0$ as $n \to \infty$, then $x \to \infty$.

[3] Instances $\mathcal{I}_1, \mathcal{I}_2$ are t-correlated if a t fraction of constraints are identical, and the rest are independent; see also [55, Definition 6.1].

[4] An *overlap gap property* exists when an inner product, for example $\frac{1}{n}\langle \mathcal{A}(\mathcal{I}_1), \mathcal{A}(\mathcal{I}_2)\rangle$, is forbidden to take values within a certain range $[a,b]$.

Given a polynomial ξ with non-negative coefficients, let $\mathscr{L}_\xi$ denote the following set of functions:

$$\mathscr{L}_\xi \stackrel{\text{def}}{=} \left\{ \mu : [0,1) \to \mathbb{R}_{\geq 0} : \mu \text{ is right-continuous, } \|\xi''\mu\|_{\mathrm{TV}[0,t]} < \infty \ \forall t \in [0,1), \text{ and } \int_0^1 \xi''(t)\mu(t)\,dt < \infty \right\}.$$

Here, $\mathrm{TV}[a,b]$ over an interval $[a,b]$ is the total variance over partitions:

$$\|f\|_{\mathrm{TV}[a,b]} \stackrel{\text{def}}{=} \sup_n \sup_{a \leq t_0 < \cdots < t_n \leq b} \sum_{i \in [n]} |f(t_i) - f(t_{i-1})|\,.$$

Furthermore, the definition of P_ξ over $\mathscr{U}$ can be naturally extended to $\mathscr{L}_\xi$, with infimum

$$\mathsf{ALG}_\xi \stackrel{\text{def}}{=} \inf_{\mu \in \mathscr{L}_\xi} \mathsf{P}_\xi(\mu)\,. \tag{1.3}$$

Since $\mathscr{L}_\xi \supseteq \mathscr{U}$, we have $\mathsf{ALG}_\xi \leq \mathsf{OPT}_\xi$. It turns out that all *overlap-concentrated* algorithms over $\mathsf{CSP}_f(\alpha)$ are with high probability obstructed to some ALG_ξ:

▶ **Theorem 5** ([55, Corollary 6.10])**.** *Choose any* even *predicate f. For all $\epsilon > 0$ there is $\alpha_0 > 0$ such that the following is true: For all $\alpha \geq \alpha_0$, for every* overlap-concentrated *algorithm $\mathcal{A}$ over $\mathsf{CSP}_f(\alpha)$, and with high probability over instances $\mathcal{I} \sim \mathsf{CSP}_f(\alpha)$ as $n \to \infty$, $\mathcal{A}$ returns an assignment to $\mathcal{I}$ with satisfying fraction* at most

$$\mathbb{E}[f] + \frac{\mathsf{ALG}_\xi + \epsilon}{\sqrt{\alpha}}\,,$$

where $\xi(s) \stackrel{\text{def}}{=} \sum_{j=1}^r \|f^{=j}\|^2 s^j$, and $\|f^{=j}\|^2$ is the Fourier weight of f at degree j.

Note that if $\mathsf{ALG}_\xi \neq \mathsf{OPT}_\xi$, then these *overlap-concentrated* algorithms are obstructed from reaching the optimal solution for typical instances of $\mathsf{CSP}_f(\alpha)$.

1.2 Statement of results

The main result of our work is that there is an *optimal* algorithm among *overlap-concentrated* algorithms over $\mathsf{CSP}_f(\alpha)$, subject to a technical condition. This matches analogous results for Ising spin glasses [1], and for multi-species spherical spin glasses [84, 48].

▶ **Assumption 1** (Proposed in [79, Remark 1.7])**.** *For any polynomial ξ with non-negative coefficients and where $\xi''(0) \neq 0$,*[5] *ALG_ξ is achieved by some $\mu_* \in \mathscr{L}_\xi$. That is, the minimizer of Equation* (1.3) *exists.*

We include Assumption 1 because our algorithm uses the minimizer μ_*. As noted in [1, journal version, Remark 2.4], Assumption 1 should not have a major impact on the algorithm's performance, since by an informal continuity argument, using a near-minimizer $\widetilde{\mu}_* \in \mathscr{L}_\xi$ instead of μ_* incurs negligible error. We further discuss this in Section 5.

▶ **Theorem 6.** *Suppose Assumption 1 holds. Choose any predicate f without linear-weight terms; i.e. $\|f^{=1}\|^2 = 0$. For all $\epsilon > 0$ there is $\alpha_0 > 0$ such that the following is true: For all $\alpha \geq \alpha_0$, there exists a* randomized *algorithm $\mathcal{A}$ such that $\mathcal{A}$ is* overlap-concentrated *over $\mathsf{CSP}_f(\alpha)$, and with high probability over* both *instances $\mathcal{I} \sim \mathsf{CSP}_f(\alpha)$ and internal randomness of $\mathcal{A}$ as $n \to \infty$, $\mathcal{A}$ returns an assignment to $\mathcal{I}$ with satisfying fraction* at least

$$\mathbb{E}[f] + \frac{\mathsf{ALG}_\xi - \epsilon}{\sqrt{\alpha}}\,,$$

where $\xi(s) \stackrel{\text{def}}{=} \sum_{j=1}^r \|f^{=j}\|^2 s^j$, and $\|f^{=j}\|^2$ is the Fourier weight of f at degree j.

[5] It is shown in [79] that ALG_ξ is not achieved by any $\mu \in \mathscr{L}_\xi$ when $\xi''(0) = 0$. See also Remark 14.

This is analogous to work of [1], which proves a similar result for mean-field spin glasses. This also extends the algorithm of [3], which studies Maximum Cut on random regular graphs. In fact, all of these algorithms are related to the Approximate Message Passing (AMP) framework, popularized by the results of [13]; see [7] for a recent survey. It was suspected that such an algorithm could be written down for $\mathsf{CSP}_f(\alpha)$ [55, 41].

One immediate impact of this result is in quantum computing. In the gate-based circuit model, many quantum circuits are *overlap-concentrated* for instances of $\mathsf{CSP}_f(\alpha)$ of depth up to $c \cdot \log n$ for some c [24]. This work implies that such quantum circuits, even up to $c \cdot \log n$ depth, cannot outperform classical algorithms on typical instances of a random CSP.

▶ **Corollary 7** (Failure of quantum advantage). *Suppose Assumption 1 holds. Choose any* even *predicate* f. *For all* $\epsilon > 0$ *there exist* $\alpha_0 > 0$ *and* $c : \mathbb{R}^{>0} \to \mathbb{R}^{>0}$ *such that the following is true: For all* $\alpha \geq \alpha_0$, *and with high probability over instances* $\mathcal{I} \sim \mathsf{CSP}_f(\alpha)$ *as* $n \to \infty$, no quantum circuit *with depth up to* $c(\alpha) \cdot \log n$ *that applies 2-local gates on a fixed, bounded-degree architecture can produce an assignment with satisfying fraction that outperforms classical algorithms by more than* $\frac{\epsilon}{\sqrt{\alpha}}$.

Even until now, the promise of quantum advantage for optimization problems has been unclear. Multiple times, improved analyses of the Quantum Approximate Optimization Algorithm (QAOA) [34] suggesting a such an advantage [35, 88, 77, 11] led to better classical algorithms [9, 45, 63, 8, 3]. This work shows that for the QAOA and many other quantum circuits up to logarithmic depth, there can be *no quantum advantage* on random CSPs with even predicates of sufficiently large clause density. Note that unlike related work on the QAOA [32, 12, 4], this rules out quantum advantage *regardless* of the presence of overlap gap phenomena.

We remark on the technical condition of *even predicates*. This can likely be removed with two steps. First, the obstruction of *overlap-concentrated* algorithms only applies to *even* predicates. New techniques to prove OGPs for spherical spin glasses [47] can possibly be ported to all Ising spin glasses; these results would immediately transfer to random instances of $\mathsf{CSP}_f(\alpha)$ via [55]. Second, the algorithm we analyze applies when predicates have no linear-weight terms; adjustments taken in [79] will likely generalize this algorithm. These steps would extend Corollary 7 to all predicates f.

Previous works noting limitations of logarithmic-depth quantum algorithms on sparse random optimization problems [32, 33, 24] also rely on the concept of *overlap gap property*. As in [1, 48], this work shows the existence of a family of random optimization problems with an *optimal* algorithm up to the OGP barrier. All of the optimal algorithms are classical, which hints that quantum algorithms that cannot pass the OGP barrier offer no additional power for classical optimization in general.

1.3 Proof overview

Our proof builds on a number of other works, especially [68, 1, 3]. We outline the proof here. Due to the page limit, some proofs in this paper are deferred to the full version.

In Section 2, we define our model of CSP and describe the algorithm. Instead of $\mathsf{CSP}_f(\alpha)$, we run the algorithm on a CSP whose associated constraint hypergraph is locally a regular *hypertree*.[6] This algorithm runs in *iterations*, where on each iteration, every vertex recomputes its value using information from adjacent vertices; we parameterize this algorithm by the functions (two per *iteration number*) that recompute each vertex's value.

[6] In the hypergraph, the variables are vertices and the constraints are hyperedges.

Table 1 List of important terms used in the analysis of message-passing algorithms for mean-field spin glasses and random CSPs. Terms in the first row asymptotically approach independent Gaussians after applying state evolution, and so constitute the *Brownian motion* (up to normalization) in the stochastic picture.

Term	**[68]**	**[1]**	**[3]**	**(our work)**
Brownian motion (asymptotically independent)	u	Δ	u	u
sum of Brownian motions	–	z	–	w
nonlinearity controller	x	x	x	x
value of spin	z	$m = f(\dots)$	z	z

We can already study the performance of the parameterized algorithm, analyzing how information propagates locally through the hypertree. In Section 3, we generalize a set of σ-algebras from [3] to describe this behavior. This allows us to analyze the correlation of a variable from the algorithm with any other variable from a particular σ-algebra. With these tools in hand, we simplify the expected performance of the parameterized algorithm, deferring some detailed calculations to the full version of this paper.

We then show how the algorithm, with the right choice of parameters, solves a stochastic optimal control problem that is dual to the minimization problem in Equation (1.3).

1. First, we show that with high probability as $\alpha \to \infty$, the intermediate values in the algorithm are asymptotically *Gaussian*. We prove this in Section 4 with an argument known as *state evolution*. Notably, this argument works for sparse models (as in [3]), and for message-passing on hypergraphs, where some values are products of Gaussians. This allows us to study quantities in the algorithm output as *discrete* stochastic processes.
2. Then, we take the number of iterations to a very large constant while proportionally decreasing the step size. Under this process, each discrete stochastic process asymptotically approximates a *continuous* stochastic process. In Section 5, we show how to choose the optimal parameters. We also show that rounding the algorithm's output to a value in $\{\pm 1\}^n$ does not meaningfully affect its performance.

This work extends [3] in a spiritually similar way that [1] extends [68]; roughly, we extend from graphs to *hypergraphs*. This requires more involved calculations and in some cases new arguments. For example, we require a more complicated proof of state evolution, since the messages in our algorithm are not Gaussian at fixed degree. Furthermore, our proof of rounding is new, since we cannot rely on spectral statistics of random *graphs*. Finally, our analysis uses an additional symmetry we call *index-regularity* (defined in Section 2) which, to the best of our knowledge, does not come up in prior work. We show that enforcing this symmetry has negligible effect (proof deferred to the full version of the paper) and use this symmetry to simplify some of our analysis.

There are a number of quantities to keep track of. We annotate the important symbols in Table 1. Following notation in [3], we typically use lowercase for a quantity in the algorithm (e.g. $u^{\ell}_{i\to a}$), uppercase with step size δ for a *discrete* stochastic process (e.g. U^{δ}_{ℓ}), and uppercase with time t for a *continuous* stochastic process (e.g. U_t).

1.4 Related work

Approximate Message Passing. Approximate Message Passing (AMP) is an algorithmic framework used to design locally computable, iterative methods for a variety of high-dimensional statistical inference and recovery tasks [61, 76, 67, 64, 10]. Many of the ideas

motivating AMP originate from the study of *belief propagation*, an iterative method rediscovered multiple times in different scientific contexts; see [65, 58] for a broader history. AMP was first introduced in the context of compressed sensing and signal recovery [29, 57], partly inspired by the use of structured priors in decoding low-density parity check codes [75, 69]. In spin glass theory, AMP is synonymous with computing the fixed points of a certain Thouless–Anderson–Palmer (TAP) free energy functional [87, 59, 90, 22, 23]. In fact, our algorithm is inspired by similar connections between the TAP equations and belief propagation [66, 65].

A beneficial feature of AMP is that one can rigorously analyze the asymptotic behavior of statistics in the framework. This is typically done through the use of *state evolution* statements [14, 13]: laws of large numbers which dictate the behavior of such statistics when the dimension in the problem approaches infinity. This has been subsequently used to compute approximate ground states of both Ising and spherical mean-field spin glasses [68, 1, 47], as well as Max-Cut on random sparse graphs [3, 19]. See the surveys of [38, 7] for further discussion on AMP and its variants. See also [78, 52, 54] for connections to the sum-of-squares hierarchy.

Ising spin glasses and CSPs. Although mean-field spin glasses with Ising spins were proposed as mathematical models of materials [82], their connection to combinatorial optimization has been observed for some time [39, 66]. More recently, a correspondence has emerged between Ising spin glasses and sparse random CSPs at large enough clause density; this started with relating the supremum of the two models [26, 80, 72]. This supremum was calculated using a novel heuristic argument known as "replica symmetry breaking" [73], and proven to be correct in [85, 70] (see also [71]). Moreover, the duality preserves a phenomenon known as *overlap gap* [44], a generalization of solution clustering that obstructs certain algorithms; see [21, 43, 42, 24, 50, 55, 49], and the expositions of [40, 46]. Notably, a strong version of overlap gap phenomena [50, 47, 48] is connected to the message-passing algorithms of [68, 1, 3], and this work. Other topics at the intersection of spin glasses and CSPs include approximating partition functions (e.g. [83]), sampling from Gibbs measures (e.g. [2, 51]), and computing thresholds for random CSPs (e.g. [27, 28]).

Quantum algorithms for classical optimization. Recent demonstrations of quantum advantage (e.g. [5, 91]) have led to increased excitement around using contemporary (i.e. NISQ [74]) quantum computers for a practically relevant problem. Although classical optimization is such a problem, quantum algorithms here are not proven to have any benefit over classical algorithms. The most popular proposal is the Quantum Approximate Optimization Algorithm (QAOA) [34]. Earlier suggestions of quantum advantage with the QAOA [35, 88, 77, 89] led to better classical algorithms [9, 45, 63, 8]; note that all of these comparisons were for very small circuit depth. Recently, [36, 11, 12] find an expression for the performance of the QAOA on spin glasses and sparse CSPs at any constant depth. For some problems, this is competitive with known classical algorithms; see [37, 16].

The QAOA is a *local* algorithm. It was noted that this feature can limit its performance [32, 33]. The authors in [24] unify definitions of quantum and classical local algorithms to show that the QAOA is obstructed by overlap gap phenomena on sparse problems. Because of this, [3] rules out quantum advantage through $\epsilon \cdot \log n$-depth QAOA on random regular Max-Cut instances of large degree (up to Assumption 1); our work generalizes this fact to CSPs with even predicates. Regardless of theory, there are many recent demonstrations of the QAOA on state-of-the-art quantum computing systems; see for example [31, 81, 62]. It is also known that the QAOA can solve satisfiability faster than brute force [17]; see also [25] for a related

protocol with a more-than-quadratic improvement. Some quantum algorithms for machine learning are known to provide no quantum advantage [86]. Very recently, [56] proposed a quantum algorithm for classical optimization, but we do not yet know if it gives a quantum advantage.

2 Local algorithms for sparse CSPs

A CSP instance has an implicit *directed* hypergraph, with variables as vertices and constraints as hyperedges.

▶ **Definition 8** (Directed hypergraph of a CSP)**.** *Consider a r-uniform CSP instance $\mathcal{I}$ with n variables and constraints $E(\mathcal{I})$, where each constraint $e \in E(\mathcal{I})$ involves r (ordered) variables in $[n]$. Then $\mathcal{I}$ is associated with the directed hypergraph with vertices $[n]$ and hyperedges $E(\mathcal{I})$; i.e. $G([n], E(\mathcal{I}))$.*

We consider a hyperedge *directed* when the involved vertices are ordered. For example, if constraint e involves vertices $\{v_1, v_2, v_3\}$ in order $[2, 3, 1]$, then the associated directed hyperedge is $[v_2, v_3, v_1]$. We denote this ordered list as ∂e. We often refer to the CSP instance by its hypergraph. The instance is fully specified by its hypergraph, the CSP function f, and the (ordered) randomness $[\epsilon_{e_1}, \ldots, \epsilon_{e_r}]$ on each constraint $e \in E(\mathcal{I})$. For each variable i, we use ∂i to denote the set of constraints i participates in.

The *factor graph* of a hypergraph $G(V, E)$ is a bipartite graph of order $|V| + |E|$, with edges between each hyperedge $e \in E$ and each vertex in ∂e. For more information on the factor graph formalism, see [60]. Let $\mathsf{B}_i(\ell)$ be the induced subgraph of G given by the ℓ-local neighborhood around vertex $i \in V$.[7]

▶ **Definition 9.** *A finite hypergraph $G(V, E)$ is (ℓ, ϵ)-*locally treelike *if the factor graph of $\mathsf{B}_i(\ell + 1)$ is isomorphic to a tree for at least $(1 - \epsilon) \cdot n$ vertices $i \in V$.*

This definition is a generalization of locally treelike graphs used in [3]; see Figure 1 for an illustration.

Every *message-passing* algorithm is contained in the model of local algorithms presented in [24]. In this model, the value associated with each variable is uniquely determined by its local neighborhood.

▶ **Definition 10** (Generic p-local algorithms, [24, Definition 3.1])**.** *Consider a randomized algorithm $\mathcal{A}$ that inputs a r-uniform hypergraph $G(V, E)$ and outputs a label in an alphabet Σ to each vertex in the graph, i.e. $\mathcal{A}(G) \in \Sigma^V$. Then $\mathcal{A}$ is* generic p-local *if for any set $S \subseteq V$, the following conditions hold:*

1. (Local distribution determination) *The joint marginal distribution of $(\mathcal{A}(G)_v)_{v \in S}$ is identical to that of $(\mathcal{A}(G')_v)_{v \in S}$, where $G' = \bigcup_{v \in S} \mathsf{B}_v(p)$.*
2. (Local independence) *$\mathcal{A}(G)_v$ is statistically independent of $\mathcal{A}(G)_{v'}$ over all vertices $v' \notin \mathsf{B}_v(2p)$.*

This model also includes the Quantum Approximate Optimization Algorithm [34] on typical instances of $\mathsf{CSP}_f(\alpha)$. In fact, any depth-p quantum circuit with a fixed architecture of bounded degree can be seen as generic p-local by including the architecture in the input hypergraph. [24] proves that for all $p \leq c(\alpha) \cdot \log n$, generic p-local algorithms are overlap-concentrated over instances of $\mathsf{CSP}_f(\alpha)$ with high probability as $n \to \infty$.

[7] Note that we define distance as the number of *hypergraph vertices* in the shortest path; vertices that share a hyperedge are distance 1 apart.

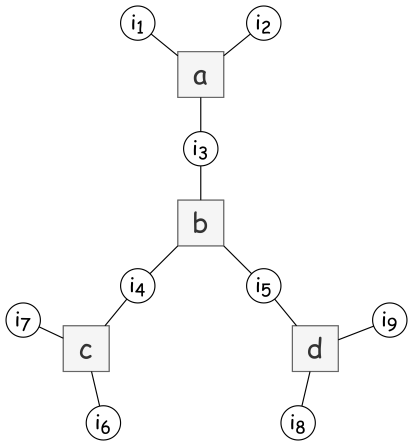


Figure 1 Factor graph of the hypergraph with vertices $\{i_1, \ldots, i_9\}$ and hyperedges $\{a, b, c, d\}$. Note that this hypergraph is locally treelike.

In our analysis, we make some convenient modifications to the input. First, since our algorithm is local and the instances are locally treelike on all but a negligible fraction of vertices[8], we analyze the algorithm on an instance which is locally a *hypertree*, i.e. a hypergraph whose factor graph is a tree. We also make the hypergraph *regular*, by embedding our instance of $\mathsf{CSP}_f(\alpha)$ in a locally treelike d-regular instance with $d = r \cdot \alpha + o_\alpha(1)$.

On hypergraphs, we need one additional property, which we call *index-regular*. Here, we mean that for each $\iota \in [r]$, each variable is used in *index* ι of a constraint the same number of times.

▶ **Definition 11** (d-index-regular CSP). *Consider a (r-uniform) CSP instance $\mathcal{I}$ with n variables and constraints $E(\mathcal{I})$, such that each constraint $e \in E(\mathcal{I})$ has associated variables $\partial e = [v_{e,1}, \ldots, v_{e,r}]$. Then $\mathcal{I}$ is* d-index-regular *if every variable $v \in [n]$ is involved in d constraints, and moreover, for each $\iota \in [r]$, v is the ιth-indexed variable in exactly $\frac{d}{r}$ constraints.*

We describe our choice of algorithm in Figure 2. This involves some notation we describe here. A function that depends on zero inputs is a constant; i.e. $A_i^0 = F_0$ and $A_{i\to a}^0 = F_{\to,0}$. For any scalar quantity that depends on vertex, a bolded version represents a vector quantity. For example, given $\partial a = [i_1, \ldots, i_r]$, $\boldsymbol{z}_{\partial a \to a}^\ell$ is the *ordered* list $[z_{i_1\to a}^\ell, \ldots, z_{i_r\to a}^\ell]$, and $\boldsymbol{z}_{\partial a}^\ell$ is the *ordered* list $[z_{i_1}^\ell, \ldots, z_{i_r}^\ell]$. Similarly, $\boldsymbol{z}^L$ is the vector in $\mathbb{R}^V$ such that $\left(\boldsymbol{z}^L\right)_i = z_i^L$.

[8] In particular, for any finite ℓ, with high probability our instances are (ℓ, ϵ)-locally treelike for some ϵ, where $\epsilon \to 0$ as $n \to \infty$.

Input: The timestep $\delta > 0$, and a d-index-regular hypergraph with girth at least $2L+2$, where $L = \lfloor \frac{1}{\delta} \rfloor$.
Parameters: The CSP function $f : \{\pm 1\}^r \to \{0,1\}$, constant $K > 0$, and nonlinearities

$$\big\{F_{\to,\ell} : \mathbb{R}^\ell \to [-K,K]\big\}_{\ell=0,\dots,L-1} \qquad \big\{F_\ell : \mathbb{R}^\ell \to [-K,K]\big\}_{\ell=0,\dots,L-1}$$

Algorithm:

1. Initialize the following:
   a. $z_i^0 \sim \mathcal{N}(0,\delta)$ for all $i \in V$ and $z_{i\to a}^0 = z_i^0$ for all $i \in \partial a$ and $a \in E$.
   b. $w_i^0 = 0$ for all $i \in V$ and $w_{i\to a}^0 = 0$ for all $i \in \partial a$ and $a \in E$.
2. For each iteration $\ell = 0, \dots, L-1$:
   a. Let $A_{i\to a}^\ell = F_{\to,\ell}\big(u_{i\to a}^1, \dots, u_{i\to a}^\ell\big)$, and let $A_i^\ell = F_\ell\big(u_i^1, \dots, u_i^\ell\big)$.
   b. For each factor $a \in E$ and involved variable $i \in \partial a$, compute:

$$w_{i\to a}^{\ell+1} = \frac{1}{\sqrt{d-1}} \cdot \sum_{b \in \partial i \setminus a} \mathsf{D}_{i;b} f(\boldsymbol{z}_{\partial b \to b}^\ell)$$
$$u_{i\to a}^{\ell+1} = w_{i\to a}^{\ell+1} - w_{i\to a}^\ell$$
$$z_{i\to a}^{\ell+1} = z_{i\to a}^0 + \sum_{s=1}^{\ell+1} A_{i\to a}^{s-1} \cdot u_{i\to a}^s$$

   c. For each variable $i \in V$, compute:

$$w_i^{\ell+1} = \frac{1}{\sqrt{d}} \cdot \sum_{b \in \partial i} \mathsf{D}_{i;b} f(\boldsymbol{z}_{\partial b \to b}^\ell)$$
$$u_i^{\ell+1} = w_i^{\ell+1} - w_i^\ell$$
$$z_i^{\ell+1} = z_i^0 + \sum_{s=1}^{\ell+1} A_i^{s-1} \cdot u_i^s$$

Output: For every $i \in V$, output 1 with probability p_i and -1 with probability $1-p_i$, where $p_i = \begin{cases} 1 & \text{if } z_i^L \geq 1, \\ \frac{1+z_i^L}{2} & \text{if } |z_i^L| < 1, \\ 0 & \text{if } z_i^L \leq -1. \end{cases}$

Figure 2 Description of the message-passing algorithm.

Our analysis makes extensive use of the partial derivative operator. For any factor a and variable $j \in \partial a$, let $\mathsf{D}_{j;a} f$ be the partial derivative of f at the *coordinate* ι such that j is the ιth coordinate of ∂a. For example, if $\partial a = [k,i,j]$ and $f(x_1,x_2,x_3) = x_1x_2 + x_2x_3$, then $\mathsf{D}_{k;a} f(x_1,x_2,x_3) = x_2$ and $\mathsf{D}_{i;a} f(x_1,x_2,x_3) = x_1 + x_3$. Note the following:

▶ Remark 12. Since the domain of f is $\{\pm 1\}^r$, f has a representation as a multilinear polynomial. As a result, for all factors b and variables $i \in \partial b$, $\mathsf{D}_{i;b} f$ is independent of the entry with the same coordinate as i in ∂b.

We additionally require the nonlinearities to satisfy the following two properties:

▶ **Property 1** (Similar node and node-to-factor evolution). *For every $\ell \geq 0$, variable i, factor $a \in \partial i$, and $m \in \mathbb{N}$, we have $\mathbb{E}[|A_i^\ell - A_{i\to a}^\ell|^m] = O_d(\frac{1}{d^{m/2}})$.*

▶ **Property 2** (Normalization of second moment). *For every $\ell \geq 0$, variable i, and factor $a \in \partial i$, we have $\mathbb{E}[(A_i^\ell)^2] = \mathbb{E}[(A_{i\to a}^\ell)^2] = \frac{\delta}{\nu_{\ell+1}}$, where $\nu_\ell \stackrel{\text{def}}{=} \frac{\xi'(\ell\delta) - \xi'((\ell-1)\delta)}{r}$, and ξ is defined as in Theorem 6.*

In Section 5, we show how to choose the nonlinearities based on Equation (1.1), and show that our choice indeed satisfies Property 1 and Property 2.

▶ Remark 13. Our analysis of the algorithm above holds even if the CSP predicate depends on factor a, so long as the absolute value of each Fourier component is fixed (this implies that all f_a are associated with the same ξ). For example, this implies that the algorithm achieves the same value for *every* choice of signs $\{\epsilon_{i,\iota}\}_{i\in[n],\iota\in[r]}$ in Definition 1.

▶ Remark 14. If $\|f^{=2}\| = 0$, the associated ξ does not satisfy $\xi''(0) \neq 0$ as in Assumption 1. We handle this case via the following argument. First, it can be shown that ALG_ξ is continuous in each coefficient of ξ using an explicit form of the Parisi functional and the fact that $\mathsf{ALG}_\xi \geq 0$ [1, Proposition 4.1]. Then, we can add a small quadratic term to f, and run the algorithm on the perturbed instance. The error produced by this perturbation may be absorbed in the ϵ of Theorem 6.

3 Independence claims

Given random variables $\{X_1, \ldots, X_k\}$, let $\sigma(X_1, \ldots, X_k) = \sigma(\{X_1, \ldots, X_k\})$ be the *σ-algebra* generated by $\{X_1, \ldots, X_k\}$. For an introduction to σ-algebra and related concepts, see for example [30].

Consider the algorithm in Figure 2. For every $0 \leq \ell \leq L$,[9] variable v, and factor $a \in \partial v$, we'll reuse the notation $\mathsf{B}_v(\ell)$ as the set of *nodes* of distance[10] at most ℓ from v in the corresponding hypergraph. We define the set $\mathsf{B}_{v\to a}(\ell)$ as the set of nodes of distance at most ℓ from v, *excluding* nodes that reach v through the factor a. We define

$$\mathcal{G}_v^\ell \stackrel{\text{def}}{=} \sigma\left(\{z_i^0 \mid i \in \mathsf{B}_v(\ell)\}\right), \qquad \mathcal{G}_{v\to a}^\ell \stackrel{\text{def}}{=} \sigma\left(\{z_i^0 \mid i \in \mathsf{B}_{v\to a}(\ell)\}\right).$$

Intuitively, these $\mathcal{G}_v^\ell$ algebras are the random variables determined by the initial Gaussian variables in the radius-ℓ neighborhood around v; here, ℓ is a *distance*. We formalize a series of information-theoretic claims using these σ-algebras, similar to [3, Lemma 2.2]. See also an illustration in Figure 3.

The first fact about independence informally says that node-to-factor messages never use information on "the other side" of the factor; once information goes across a, it never comes back.

▶ **Lemma 15.** *For every $0 \leq \ell_1, \ell_2 \leq L$, distinct variables i, j, and factors $a \in \partial i, b \in \partial j$, if every path of length at most $2L$ from i to j goes through a and b, then $\mathcal{G}_{i\to a}^{\ell_1}$ and $\mathcal{G}_{j\to b}^{\ell_2}$ are independent.*

[9] Recall that because the instance has girth at least $2L+2$, all local neighborhoods of radius up to L are treelike.

[10] Once again, we consider *distance* of i and j as the number of *nodes* required to pass from i to j; the factors are not counted.

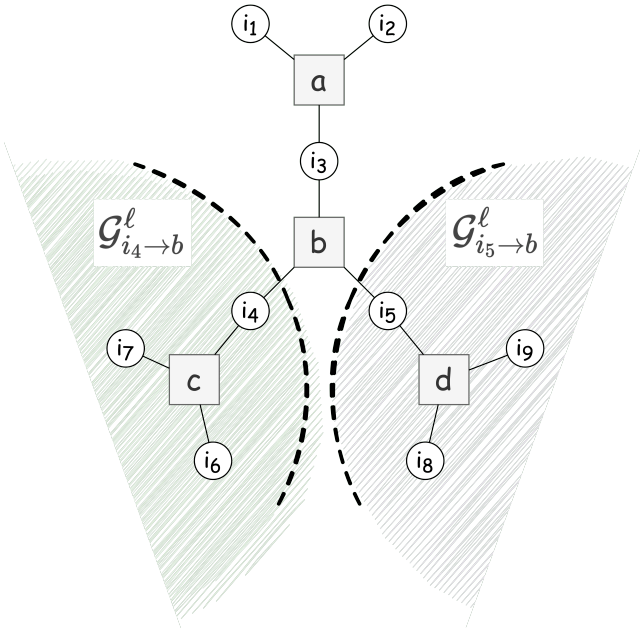


Figure 3 An illustration of σ-algebras on the hypertree. Since the path from i_4 to i_5 goes through factor b, $\mathcal{G}^{\ell}_{i_4\to b}$ and $\mathcal{G}^{\ell}_{i_5\to b}$ are independent by Lemma 15. In addition, $\mathcal{G}^{\ell+1}_{i_3\to a} = \sigma\left(\mathcal{G}^{0}_{i_3} \bigcup \mathcal{G}^{\ell}_{i_4\to b} \bigcup \mathcal{G}^{\ell}_{i_5\to b}\right)$ by Lemma 16.

Lemma 15 holds even if the factors are the same, i.e. $a = b$. We refer to this case as Lemma 15 *around* factor a.

The next fact concerns the geometry of local neighborhoods. It allows us to decompose each σ-algebra into σ-algebras with distance reduced by one, which is handy for induction.

▶ **Lemma 16.** *For every $1 \le \ell \le L$, variable i, and factor $a \in \partial i$, we have*

$$\mathcal{G}^{\ell}_{i} = \sigma\left(\mathcal{G}^{\ell}_{i\to a} \cup \left(\bigcup_{j\in\partial a\setminus i} \mathcal{G}^{\ell-1}_{j\to a}\right)\right), \qquad \mathcal{G}^{\ell}_{i\to a} = \sigma\left(\mathcal{G}^{0}_{i} \cup \left(\bigcup_{b\in\partial i\setminus a}\ \bigcup_{v\in\partial b\setminus i} \mathcal{G}^{\ell-1}_{v\to b}\right)\right).$$

The next statement sorts quantities appearing in the algorithm (Figure 2) into appropriate σ-algebras. It proves the intuition that messages at iteration number ℓ are determined by the local neighborhood of distance at most ℓ:

▶ **Lemma 17.** *For every $1 \le \ell \le L$, variable i, and factor a, we have $u^{\ell}_{i\to a}, w^{\ell}_{i\to a}, z^{\ell}_{i\to a} \in \mathcal{G}^{\ell}_{i\to a}$ and $u^{\ell}_{i}, w^{\ell}_{i}, z^{\ell}_{i} \in \mathcal{G}^{\ell}_{i}$.*

Lemma 17 implies that the ℓth node message at variable i is determined by the initial Gaussian variables in the radius-ℓ neighborhood around i. Note that the node-to-factor message $u^{\ell}_{i\to a}$ is *not* influenced by Gaussians at nodes that pass through a to reach i.

Next, we prove that messages at node i and iteration number $\ell+1$ are uncorrelated with the initial Gaussians in the radius-ℓ neighborhood around i. Put another way, the "extra randomness" at distance $\ell+1$ removes any correlation with $\mathcal{G}^{\ell}_{i}$. To do this, we show that the messages $\{u^{\ell}_{j\to b}\}_{j\in\partial b\setminus i}$ are uncorrelated with each other, conditioned on $\mathcal{G}^{\ell}_{i}$.[11]

[11] This is subtle. Although $\{u^{\ell}_{j\to b}\}_{j\in\partial b\setminus i}$ are independent by Lemma 15 around factor b, independence does not imply *conditional* independence.

▶ **Lemma 18.** *Choose any variable i, factor $b \in \partial i$, and $1 \le \ell \le L$. Choose also $s_j \ge 0$ for every $j \in \partial b \backslash i$. Then*

$$\mathbb{E}\left[\prod_{j\in\partial b\backslash i}(u^{\ell}_{j\to b})^{s_j}\;\middle|\;\mathcal{G}^{\ell}_i\right] = \prod_{j\in\partial b\backslash i}\mathbb{E}\left[(u^{\ell}_{j\to b})^{s_j}\;\middle|\;\mathcal{G}^{\ell}_i\right].$$

▶ **Lemma 19.** *Choose any variable i, factor $a \in \partial i$, and $0 \le \ell \le L-1$. Then $\mathbb{E}\left[u^{\ell+1}_{i\to a}\mid \mathcal{G}^{\ell}_i\right] = 0$ and $\mathbb{E}\left[u^{\ell+1}_{i}\mid \mathcal{G}^{\ell}_i\right] = 0$.*

▶ **Corollary 20.** *Fix any variable i, factor $b \in \partial i$, and $0 \le \ell \le L$. Then $\mathbb{E}\left[\mathsf{D}_{i;b}f(\boldsymbol{z}^{\ell}_{\partial b\to b}) \mid \mathcal{G}^{\ell}_i\right] = \mathsf{D}_{i;b}f(\boldsymbol{z}^{\ell-1}_{\partial b\to b})$ if $\ell \ge 1$ and 0 otherwise.*

3.1 Simplifying the main sum

What is the average value achieved by the algorithm? By linearity, we can look at the value of any factor a; i.e. $\mathbb{E}[f(\boldsymbol{z}^{L}_{\partial a})]$. Note that $\mathbb{E}[f(\boldsymbol{z}^{0}_{\partial a})] = f(0,\dots,0) = \mathbb{E}[f]$, and

$$\begin{aligned}\mathbb{E}[f(\boldsymbol{z}^{L}_{\partial a})] &= \mathbb{E}[f(\boldsymbol{z}^{0}_{\partial a})] + \sum_{\ell=0}^{L-1}\left(\mathbb{E}[f(\boldsymbol{z}^{\ell+1}_{\partial a})] - \mathbb{E}[f(\boldsymbol{z}^{\ell}_{\partial a})]\right)\\ &= \mathbb{E}[f] + \sum_{\ell=0}^{L-1}\left(\mathbb{E}[f(\boldsymbol{z}^{\ell+1}_{\partial a})] - \mathbb{E}[f(\boldsymbol{z}^{\ell}_{\partial a})]\right).\end{aligned}$$

It suffices to compute the difference $\mathbb{E}[f(\boldsymbol{z}^{\ell+1}_{\partial a})] - \mathbb{E}[f(\boldsymbol{z}^{\ell}_{\partial a})]$ for every $0 \le \ell \le L-1$. We use the Fourier expansion of f to express this as a sum of differences of monomials.

First, we simplify terms of the form $u_i^{s_i}Y_i^{s_i-1}\prod_{v\in\partial a\backslash i}Y_v^{s_v}$, which we encounter along the way. This relies on the following consequence of the monotone class theorem:

▶ **Lemma 21** (e.g. [30, Theorem 5.2.2], [3, Lemma 2.3])**.** *Let $\{\mathcal{F}_1,\dots,\mathcal{F}_m,\mathcal{A}\}$ be a set of σ-algebras so that $\mathcal{F} \stackrel{\text{def}}{=} \sigma(\mathcal{F}_1,\dots,\mathcal{F}_m) \subseteq \mathcal{A}$. Choose a random variable $X \in \mathcal{A}$,[12] and suppose for any $Y_1 \in \mathcal{F}_1,\dots,Y_m \in \mathcal{F}_m$, $\mathbb{E}[XY_1\dots Y_m] = 0$ when the expectation exists. Then $\mathbb{E}[XY] = 0$ (when the expectation exists) for every $Y \in \mathcal{F}$.*

As a result, to show an expectation is zero, we may replace a variable in σ-algebra $\mathcal{F}$ with a product of variables from σ-algebras that cover $\mathcal{F}$.

▶ **Lemma 22** (Generalization of [3, Lemma 2.5])**.** *Choose factor a, integers $s_v \ge 0$ for $v \in \partial a$, and let $Y_v^{s_v} \in \mathcal{G}_v^{s_v}$. Let $i \in \partial a$ be such that $s_i + 1 \ge s_v$ for all $v \in \partial a$. Then*

$$\mathbb{E}\left[u_i^{s_i+1}\prod_{v\in\partial a}Y_v^{s_v}\right] = \frac{1}{\sqrt{d}}\,\mathbb{E}\left[\left(\mathsf{D}_{i;a}f(\boldsymbol{z}^{s_i}_{\partial a\to a}) - \mathbb{1}_{[s_i\ge 1]}\mathsf{D}_{i;a}f(\boldsymbol{z}^{s_i-1}_{\partial a\to a})\right)\prod_{v\in\partial a}Y_v^{s_v}\right],$$

where $\mathbb{1}_{[x\ge y]} = 1$ if $x \ge y$ and 0 otherwise.

▶ **Proposition 23.** *Fix $0 \le \ell \le L-1$, factor a, and $S \subseteq \partial a$. Then*

$$\mathbb{E}\left[\prod_{j\in S}z_j^{\ell+1}\right] - \mathbb{E}\left[\prod_{j\in S}z_j^{\ell}\right] = \sum_{i\in S}\mathbb{E}\left[A_i^{\ell}u_i^{\ell+1}\cdot\prod_{j\in S\backslash i}z_j^{\ell}\right] + O_d\left(\frac{1}{d}\right). \tag{3.1}$$

[12] $X \in \mathcal{A}$ means that $\sigma(X) \subseteq \mathcal{A}$.

▶ **Corollary 24.** *Choose $0 \le \ell \le L-1$ and factor a. Then*

$$\mathbb{E}\left[f(\boldsymbol{z}_{\partial a}^{\ell+1})\right] - \mathbb{E}\left[f(\boldsymbol{z}_{\partial a}^{\ell})\right] = \sum_{i \in \partial a} \mathbb{E}\left[A_i^\ell u_i^{\ell+1} \cdot \mathsf{D}_{i;a} f(\boldsymbol{z}_{\partial a}^{\ell})\right] + O_d\left(\frac{1}{d}\right).$$

We simplify this expression further, assuming the algorithm's node and node-to-factor nonlinearities are similar (Property 1). We choose nonlinearities exhibiting this property in Section 5.

▶ **Proposition 25.** *Assume Property 1. Let $0 \le \ell \le L-1$, a be a factor, and $i \in \partial a$. Then*

$$\frac{1}{\alpha n}\sum_{a \in E}\left(\mathbb{E}\left[f(\boldsymbol{z}_{\partial a}^{\ell+1})\right] - \mathbb{E}\left[f(\boldsymbol{z}_{\partial a}^{\ell})\right]\right) = \frac{r}{\sqrt{d}} \cdot \frac{1}{n}\sum_{i \in V} \mathbb{E}\left[A_i^\ell (u_i^{\ell+1})^2\right] + O_d\left(\frac{1}{d}\right).$$

By these propositions, we get the following:

▶ **Theorem 26.** *Assume Property 1. Fix a constant $L \ge 1$. Then*

$$\frac{1}{\alpha n}\sum_{a \in E} \mathbb{E}\left[f(\boldsymbol{z}_{\partial a}^{L})\right] = \mathbb{E}[f] + \frac{r}{\sqrt{d}} \cdot \sum_{\ell=0}^{L-1}\left(\frac{1}{n}\sum_{i \in V} \mathbb{E}\left[A_i^\ell (u_i^{\ell+1})^2\right]\right) + O_d\left(\frac{1}{d}\right).$$

4 State evolution for message-passing on hypergraphs

So far, we've shown that the value of the algorithm takes a simple form (i.e. Theorem 26). From here, we prove a *state evolution* statement: with some weak conditions on the nonlinearities (Property 2), we analyze the asymptotic behavior of statistics of the algorithm. State evolution as applied to AMP first appeared in [13]; this was inspired by a proof in a paper computing the fixed points of the TAP equations for the Sherrington–Kirkpatrick model [14]. Since then, state evolution has been used in AMP algorithms applied to a variety of tasks; see for example [38].

Here, we consider functions of the messages $\{u_{i\to a}^\ell\}_{\ell \in [L]}$ (or $\{u_i^\ell\}_{\ell \in [L]}$) that are *pseudo-Lipschitz*:

▶ **Definition 27.** *Fix $s > 0$. We say that a given function $\psi : \mathbb{R}^s \to \mathbb{R}$ is* pseudo-Lipschitz *if there exists a constant $C > 0$ such that*

$$|\psi(\boldsymbol{x}) - \psi(\boldsymbol{y})| \le C \cdot \left(1 + \|\boldsymbol{x}\| + \|\boldsymbol{y}\|\right) \cdot \|\boldsymbol{x} - \boldsymbol{y}\|,$$

for every $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^s$.

As $d \to \infty$, the expected value of a pseudo-Lipschitz function ψ with messages $\{u_{i\to a}^\ell\}_{\ell \in [L]}$ (or $\{u_i^\ell\}_{\ell \in [L]}$) as input approaches the expected value of ψ with independent Gaussians as input.

▶ **Definition 28.** *For any predicate $f : \{\pm 1\}^r \to \{0,1\}$, let ξ be defined as in Theorem 6. For any $\ell \ge 1$, define the quantity*[13]

$$\nu_\ell \stackrel{\text{def}}{=} \frac{\xi'(\ell\delta) - \xi'((\ell-1)\delta)}{r}.$$

[13] Note this is the same parameter ν_ℓ in Property 2.

▶ **Theorem 29.** *Assume Property 2. Then for any pseudo-Lipschitz function $\psi : \mathbb{R}^\ell \to \mathbb{R}$, variable i, factor $a \in \partial i$, $1 \le \ell \le L$, we have*

$$\mathbb{E}\left[\psi(u^1_{i\to a}, \dots, u^\ell_{i\to a})\right] = \mathbb{E}\left[\psi(U_1, \dots, U_\ell)\right] + o_d(1)\,, \tag{4.1}$$

$$\mathbb{E}\left[\psi(u^1_{i}, \dots, u^\ell_{i})\right] = \mathbb{E}\left[\psi(U_1, \dots, U_\ell)\right] + o_d(1)\,, \tag{4.2}$$

where $U_s \sim \mathcal{N}(0, \nu_s)$ independently.

Theorem 29 is inspired by the state evolution statement in [3]. These statements are for *sparse* problems, and so do not require a correction term to ensure variables remain Gaussian (often named the *Onsager correction*). In [3], the messages $\{u^\ell_{i\to a}\}_{\ell\in[L]}$ are Gaussian even at fixed degree! In our algorithm, this is not the case for arbitrary predicates f; we must take additional care showing that these quantities are close to Gaussian.

We defer the proof of Theorem 29 to the full version of the paper. As a quick overview of the proof, we first establish that given Property 2, $u^\ell_{i\to a}$ is *close* to the Gaussian $\mathcal{N}(0, \nu_\ell)$ when $d \to \infty$, independently of $\mathcal{G}_i^{\ell-1}$. Then, we show that pseudo-Lipschitz functions are smooth enough so that an input $u^\ell_{i\to a}$ may be exchanged with an independent Gaussian input, and use this argument inductively to finish the proof.

Given Theorem 29, we may convert each spin $\{z_i^\ell\}_{0\le\ell\le L}$ to a certain *Gaussian process*. In Section 5, we do exactly this, and choose the nonlinearities $\{A_i^\ell\}_{0\le\ell\le L}$ to maximize the expression in Theorem 26 (subject to $|z_i^L| \le 1$).

5 Choosing the nonlinearities

We now design the nonlinearities to maximize Theorem 26. In light of Section 4, we consider the stochastic optimal control problem of [1]. Define $\xi(s) \stackrel{\text{def}}{=} \sum_{j=1}^r \|f^{=j}\|^2 s^j$. Consider any $\mu \in \mathscr{L}_\xi$, and Φ^μ chosen as in Equation (1.1). Define the stochastic differential equation[14]

$$\mathrm{d}X_t = \xi''(t)\mu(t)\Phi^\mu_x(t, X_t)\mathrm{d}t + \sqrt{\xi''(t)}\mathrm{d}B_t\,, \quad X_0 = 0\,, \tag{5.1}$$

along with associated martingale

$$Z_t = \int_0^t \sqrt{\xi''(s)}\Phi^\mu_{xx}(s, X_s)\mathrm{d}B_s\,. \tag{5.2}$$

We now crucially use Assumption 1. Let $\mu_* \in \mathscr{L}$ be the minimizer of Equation (1.3).

▶ **Lemma 30** ([1, Theorem 3])**.** *Suppose Assumption 1 holds. Then for every $\epsilon > 0$, there exists $\eta > 0$ such that*

$$\int_0^{1-\eta} \xi''(t)\,\mathbb{E}[\Phi^{\mu_*}_{xx}(t, X_t)]\mathrm{d}t \ge \mathsf{ALG}_\xi - \epsilon\,.$$

For mean-field spin glasses, Parisi predicted that the minimizer of $\inf_{\mu\in\mathscr{U}} \mathsf{P}_\xi(\mu)$ led to the correct description of the ground state. While this was true, uniqueness (Theorem 2) required a stochastic representation of the minimization problem. Later, [68, 1] formulated an algorithm whose optimal value can be written as a stochastic optimal control problem dual to Equation (1.3). Surprisingly, the study of overlap gap phenomena is also linked to Equation (1.3) [50, 55], so this algorithm is optimal among a broad family.[15]

[14] Here, $(B_t)_{t\ge0}$ is standard Brownian motion.
[15] This is the family of *overlap-concentrated* algorithms; see Definition 4.

In a sense, the optimal message-passing algorithm updates its local information in a way concordant with the Parisi minimizer μ_*, *assuming* that μ_* exists. Nonetheless, even if Assumption 1 is false, we informally argue that this algorithm succeeds. Since Φ^μ_x and Φ^μ_{xx} are continuous in μ (e.g., [20, Theorem 4]), using a near-minimizer $\widetilde{\mu}_*$ of Equation (1.3) changes the value in Lemma 30 by a negligible amount, which can be absorbed into the choice of ϵ. Furthermore, $\widetilde{\mu}_*$ can be precomputed, since finding $\widetilde{\mu}_*$ is a task required only once per *CSP predicate* f, not once per run of the algorithm. Mathematizing this argument, however, is a valuable open problem; see [1, Section 6].

In this section, we prove that the algorithm achieves a value affine to Lemma 30 after taking two sets of limits.

5.1 A finite difference equation

Consider discrete versions of Equations (5.1) and (5.2). Choose $\delta > 0$, and consider independent *standard* Gaussians $\{B_1, B_2, \dots\}$. Then consider the finite difference

$$X^\delta_{\ell+1} - X^\delta_\ell = \xi''(\delta\ell)\mu_*(\delta\ell)\Phi^{\mu_*}_x(\delta\ell, X^\delta_\ell)\cdot\delta + \sqrt{\xi'(\delta(\ell+1)) - \xi'(\delta\ell)}\cdot B_{\ell+1} \qquad X^\delta_0 = 0\,, \tag{5.3}$$

$$Z^\delta_{\ell+1} - Z^\delta_\ell = \sqrt{\xi'(\delta(\ell+1)) - \xi'(\delta\ell)}\cdot\Phi^{\mu_*}_{xx}(\delta\ell, X^\delta_\ell)\cdot B_{\ell+1} \qquad Z^\delta_0 = 0\,. \tag{5.4}$$

The variables are named suggestively. By Section 4, the messages $u^\ell_{i\to a}, u^\ell_i$ behave as independent Gaussians as $d\to\infty$, which we use to imitate B_ℓ above. To this end, we define the sequences

$$x^{\ell+1}_i = x^\ell_i + \xi''(\delta\ell)\mu_*(\delta\ell)\Phi^{\mu_*}_x(\delta\ell, x^\ell_i)\cdot\delta + \sqrt{r}\cdot u^{\ell+1}_i \qquad x^0_i = 0\,,$$
$$x^{\ell+1}_{i\to a} = x^\ell_{i\to a} + \xi''(\delta\ell)\mu_*(\delta\ell)\Phi^{\mu_*}_x(\delta\ell, x^\ell_{i\to a})\cdot\delta + \sqrt{r}\cdot u^{\ell+1}_{i\to a} \qquad x^0_{i\to a} = 0\,.$$

We show that in the limit $d\to\infty$, the messages $\{u^\ell_i, u^\ell_{i\to a}\}$ behave as stochastic input to Equations (5.3) and (5.4). Note that $\Phi^\mu_x(t,x)$ and $\Phi^\mu_{xx}(t,x)$ are Lipschitz in x, since all x-derivatives are of $\Phi^\mu_x(t,x)$ are bounded:

▶ **Lemma 31** ([1, Lemma 6.4])**.** *For every $\mu\in\mathscr{L}$, $\epsilon>0$, and $j\ge 2$, there exists $K = K(\mu,j,\epsilon) > 0$ such that the jth derivative in x of Φ^μ is within $[-K,K]$ for all $t\in[0,1-\epsilon]$ and $x\in\mathbb{R}$. In other words, $|\frac{\partial^j}{\partial x^j}\Phi^\mu(t,x)|\le K$ for all $t\in[0,1-\epsilon]$ and $x\in\mathbb{R}$.*

By induction and using Lemma 31, x^ℓ_i and $\Phi^{\mu_*}_{xx}(t,x^\ell_i)$, and $x^\ell_{i\to a}$ and $\Phi^{\mu_*}_{xx}(t,x^\ell_{i\to a})$ are Lipschitz in the messages $\{u^s_i\}_{s\le\ell}$ and $\{u^s_{i\to a}\}_{s\le\ell}$, respectively. So we can apply Theorem 29:

▶ **Proposition 32.** *Assume Property 2. Choose $1\le\ell\le L$ and any variable i. Then for every integer $k\ge 1$, we have*

$$\mathbb{E}[\Phi^{\mu_*}_{xx}(\delta\ell, x^\ell_i)^k] = \mathbb{E}[\Phi^{\mu_*}_{xx}(\delta\ell, X^\delta_\ell)^k] + o_d(1)\,.$$

We now choose the nonlinearities:

$$A^\ell_i = \Phi^{\mu_*}_{xx}(\delta\ell, x^\ell_i)\cdot\sqrt{\frac{\delta}{\nu_{\ell+1}\cdot\mathbb{E}[\Phi^{\mu_*}_{xx}(t,x^\ell_i)^2]}}\,, \tag{5.5}$$

$$A^\ell_{i\to a} = \Phi^{\mu_*}_{xx}(\delta\ell, x^\ell_{i\to a})\cdot\sqrt{\frac{\delta}{\nu_{\ell+1}\cdot\mathbb{E}[\Phi^{\mu_*}_{xx}(t,x^\ell_{i\to a})^2]}}\,. \tag{5.6}$$

These nonlinearities behave as we promised earlier. First of all, by Lemma 31, the nonlinearities are uniformly bounded for all d (presumed in Figure 2). Furthermore, by construction, the second moment of the nonlinearities satisfy Property 2. Finally, the nonlinearities do satisfy Property 1:

▶ **Proposition 33.** *Equations* (5.5) *and* (5.6) *satisfy Property 1. That is, for every* $m \in \mathbb{N}$, *we have* $\mathbb{E}[|A_i^\ell - A_{i\to a}^\ell|^m] = O_d(\frac{1}{d^{m/2}})$.

Note that with this choice of nonlinearities, the value of the spin $\boldsymbol{z}^\ell$ differs slightly from Equation (5.4) in the large degree limit. By Figure 2, the spin value evolves as

$$\widetilde{Z}_{\ell+1}^\delta - \widetilde{Z}_\ell^\delta = B_{\ell+1} \cdot \frac{\sqrt{\delta} \cdot \Phi_{xx}^{\mu_*}(\delta\ell, X_\ell^\delta)}{\sqrt{\mathbb{E}[\Phi_{xx}^{\mu_*}(\delta\ell, X_\ell^\delta)^2]}}, \quad \widetilde{Z}_0^\delta \sim N(0, \delta)\,.$$

However, the difference between Z_ℓ^δ and $\widetilde{Z}_\ell^\delta$ is inconsequential for small enough δ:

▶ **Proposition 34** ([1, Equation 5.7])**.** *Suppose Assumption 1 holds. Then there exist* $\delta_0, C > 0$ *such that for every* $\delta < \delta_0$ *and* $\ell < \frac{1}{\delta}$, *we have*

$$\left| \frac{\xi'((\ell+1)\delta) - \xi'(\ell\delta)}{\delta} \cdot \mathbb{E}[\Phi_{xx}^{\mu_*}(\delta\ell, X_\ell^\delta)^2] - 1 \right| \le C\sqrt{\delta}\,.$$

5.2 Achieving the extended Parisi value

Here we show that Theorem 26 achieves a value affine to ALG_ξ, the infimum of a Parisi minimization problem. First, note that the expectation $\mathbb{E}[A_i^\ell(u_i^{\ell+1})^2]$ can be simplified at large degree:

▶ **Proposition 35.** *For every variable* i *and* $0 \le \ell \le L-1$, *we have*

$$\left|\mathbb{E}\left[A_i^\ell(u_i^{\ell+1})^2\right] - \nu_{\ell+1} \cdot \mathbb{E}\left[A_i^\ell\right]\right| = O_d\left(\frac{1}{\sqrt{d}}\right).$$

After the large degree limit, we take a second limit sending the timestep to 0; i.e. $\delta \to 0$ or $\frac{1}{\delta} \to \infty$. As a result, the finite difference describing $x_i^\ell, x_{i\to a}^\ell$ converges to Equation (5.1), and the finite difference describing $z_i^\ell, z_{i\to a}^\ell$ converges to Equation (5.2). We bound the error when making this approximation, using a statement from [1]:[16]

▶ **Proposition 36** ([1, Proposition 5.3], see also [3, Proposition 4.8])**.** *Suppose Assumption 1 holds. Then there exists a coupling between* $(X_t, Z_t)_{t\ge 0}$ *and* $(X_\ell^\delta, \widetilde{Z}_\ell^\delta)_{\ell\in\mathbb{N}}$ *and a constant* $C > 0$ *such that for every small enough* δ *and* $\ell \le \frac{1}{\delta}$, *we have* $\mathbb{E}[|X_{\delta\ell} - X_\ell^\delta|^2] \le C\delta$ *and* $\mathbb{E}[|Z_{\delta\ell} - \widetilde{Z}_\ell^\delta|^2] \le C\delta$.

We are now in a position to simplify Theorem 26:

▶ **Theorem 37.** *Suppose Assumption 1 holds. Then for all* $\epsilon > 0$, *there exist* $d_0 \in \mathbb{N}$ *and* $\delta_0 > 0$ *such that for all* $d \ge d_0$ *and* $0 < \delta \le \delta_0$,

$$\frac{1}{\alpha n}\sum_{a\in E}\mathbb{E}\left[f(\boldsymbol{z}_{\partial a}^L)\right] \ge \mathbb{E}[f] + \frac{\mathsf{ALG}_\xi - \epsilon}{\sqrt{\alpha}}\,.$$

[16] The curious reader may refer to Table 1 to convert between the notation here and in [1].

Proof. Recall that $\alpha = \frac{d}{r}$ is the clause density. By Theorem 26 and Proposition 35, we have

$$\frac{1}{\alpha n}\sum_{a\in E}\mathbb{E}\left[f(\boldsymbol{z}^L_{\partial a})\right] = \mathbb{E}[f] + \frac{r}{\sqrt{d}}\cdot\sum_{\ell=0}^{L-1}\left(\frac{1}{n}\sum_{i\in V}\nu_{\ell+1}\cdot\mathbb{E}\left[A_i^\ell\right]\right) + O_d\left(\frac{1}{d}\right).$$

Let $\Delta_{\ell,i} \stackrel{\text{def}}{=} \nu_{\ell+1}\cdot\mathbb{E}[A_i^\ell]$. At large enough d, Proposition 32 and Proposition 34 imply the following for all i:[17]

$$\Delta_{\ell,i} = \nu_{\ell+1}\cdot\mathbb{E}[\Phi^{\mu_*}_{xx}(\delta\ell, X^\delta_\ell)]\cdot\sqrt{r}(1+O_{\frac{1}{\delta}}(\sqrt{\delta})) + o_d(1).$$

By definition of $\nu_{\ell+1}$,

$$\Delta_{\ell,i} = \frac{\xi'(\delta(\ell+1))-\xi'(\delta\ell)}{\sqrt{r}}\cdot\mathbb{E}[\Phi^{\mu_*}_{xx}(\delta\ell, X^\delta_\ell)]\cdot(1+O_{\frac{1}{\delta}}(\sqrt{\delta})) + o_d(1).$$

By Proposition 36 and Lipschitzness of $\Phi^{\mu_*}_{xx}(t,x)$ in x, using the continuous $X_{\delta\ell}$ introduces multiplicative error:

$$\Delta_{\ell,i} = \frac{\xi'(\delta(\ell+1))-\xi'(\delta\ell)}{\sqrt{r}}\cdot\mathbb{E}[\Phi^{\mu_*}_{xx}(\delta\ell, X_{\delta\ell})]\cdot(1+O_{\frac{1}{\delta}}(\sqrt{\delta}))^2 + o_d(1).$$

Putting it all together,

$$\begin{aligned}
&\frac{1}{\alpha n}\sum_{a\in E}\mathbb{E}\left[f(\boldsymbol{z}^L_{\partial a})\right]\\
&= \mathbb{E}[f] + \frac{r}{\sqrt{d}}\cdot\sum_{\ell=0}^{L-1}\left(\frac{1}{n}\sum_{i\in V}\Delta_{\ell,i}\right) + O_d\left(\frac{1}{d}\right)\\
&= \mathbb{E}[f] + \sqrt{\frac{r}{d}}\cdot\sum_{\ell=0}^{L-1}(\xi'(\delta(\ell+1))-\xi'(\delta\ell))\cdot\mathbb{E}[\Phi^{\mu_*}_{xx}(\delta\ell, X_{\delta\ell})]\cdot(1+O_{\frac{1}{\delta}}(\sqrt{\delta}))^2 + o_d\left(\frac{1}{\sqrt{d}}\right)\\
&= \mathbb{E}[f] + \frac{1}{\sqrt{\alpha}}\cdot\sum_{\ell=0}^{L-1}(\xi'(\delta(\ell+1))-\xi'(\delta\ell))\cdot\mathbb{E}[\Phi^{\mu_*}_{xx}(\delta\ell, X_{\delta\ell})]\cdot(1+o_{\frac{1}{\delta}}(1)) + o_d\left(\frac{1}{\sqrt{d}}\right).
\end{aligned}$$

Applying Lemma 30 with large enough d and small enough δ proves the result. ◀

5.3 Rounding

We have nearly achieved our objective. However, the entries of $\boldsymbol{z}^L$ are *continuous*, not in $\{\pm 1\}$. We account for this by *rounding* each entry of $\boldsymbol{z}^L$ without significant impact to the value of the algorithm.

Let $\mathrm{trnc}:\mathbb{R}\to[-1,1]$ be the truncation function, defined by

$$\mathrm{trnc}(x) = \begin{cases} -1 & \text{if } x < -1,\\ x & \text{if } -1 \le x \le 1,\\ 1 & \text{if } x > 1.\end{cases}$$

We use a randomized rounding procedure R that acts independently on each vertex i: it outputs $+1$ at vertex i with probability $\frac{1+\mathrm{trnc}(z_i^L)}{2}$, and -1 with probability $\frac{1-\mathrm{trnc}(z_i^L)}{2}$. The

[17] Note that an error bound in Proposition 32 could improve the scaling beyond $o_d(1)$.

average value of the algorithm is unchanged when replacing $\mathrm{trnc}(\boldsymbol{z}^L)$ with $R(\boldsymbol{z}^L)$.[18] This is because averaging over the randomness in R, we have for every set $S \subseteq V$,

$$\mathbb{E}\left[\prod_{i\in S} R(z_i^L)\right] = \prod_{i\in S} \mathrm{trnc}(z_i^L)\,.$$

We also compare the average value of the algorithm when replacing $\boldsymbol{z}^L$ with $\mathrm{trnc}(\boldsymbol{z}^L)$; i.e. the average value of $\mathbb{E}[f(\mathrm{trnc}(\boldsymbol{z}_{\partial a}^L))] - \mathbb{E}[f(\boldsymbol{z}_{\partial a}^L)]$ across factors a. Intuitively, trnc has no effect because each entry of $\boldsymbol{z}^L$ is almost surely bounded by 1 after taking the limits $d \to \infty$ and $\delta \to 0$:

▶ **Lemma 38** ([1, Lemma 6.2 and 6.5]). *Suppose Assumption 1 holds with minimizer $\mu_* \in \mathscr{L}$, and consider the martingale Z_t in Equation (5.2) corresponding to μ_*. Then for all $t \in [0,1]$, $|Z_t| \le 1$ almost surely.*

For each factor a, the difference $\mathbb{E}[f(\mathrm{trnc}(\boldsymbol{z}_{\partial a}^L))] - \mathbb{E}[f(\boldsymbol{z}_{\partial a}^L)]$ can be split into at most 2^r terms of the form

$$\mathbb{E}\left[\left(\prod_{i\in S}\mathrm{trnc}(z_i^L)\right) - \left(\prod_{i\in S} z_i^L\right)\right],$$

for some set $S \subseteq \partial a$ where $|S| \ge 2$ (since f has no linear terms). We show that each of these terms is at most $\delta \cdot O_d(\frac{1}{\sqrt{d}})$, where δ is the timestep of the algorithm.

Arbitrarily *order* the elements of set S; then the above term can be written as

$$\begin{aligned}&\mathbb{E}\left[\left(\prod_{i\in S}\mathrm{trnc}(z_i^L)\right) - \left(\prod_{i\in S} z_i^L\right)\right]\\ &= \sum_{\kappa=1}^{|S|}\mathbb{E}\left[\left(\prod_{\iota=1}^{\kappa-1}\mathrm{trnc}(z_{S[\iota]}^L)\right)\left(\mathrm{trnc}(z_{S[\kappa]}) - z_{S[\kappa]}\right)\left(\prod_{\iota=\kappa+1}^{|S|} z_{S[\iota]}^L\right)\right].\end{aligned}$$

We show that the average value of such terms across factors a is negligible with d as $n \to \infty$.

▶ **Proposition 39.** *For every integer $L \ge 1$, clause a, and disjoint subsets $T_1, T_2 \subseteq \partial a$ such that $T_1 \cup T_2 \ne \varnothing$, we have*

$$\mathbb{E}\left[\left(\Big(\prod_{i\in T_1} z_i^L\Big)\prod_{i\in T_2}\mathrm{trnc}(z_i^L) - \Big(\prod_{i\in T_1} z_{i\to a}^L\Big)\prod_{i\in T_2}\mathrm{trnc}(z_{i\to a}^L)\right)^2\right] = O_d\left(\frac{1}{d}\right).$$

▶ **Proposition 40.** *Fix an integer $L \ge 1$ and variable i. For each $a \in \partial i$, fix disjoint subsets $T_{1,a}, T_{2,a} \subseteq \partial a \backslash i$ with $T_{1,a} \cup T_{2,a} \ne \varnothing$. Then*

$$\left|\mathbb{E}\left[(z_i^L - \mathrm{trnc}(z_i^L))\cdot\frac{1}{d}\sum_{a\in\partial i}\prod_{j\in T_{1,a}} z_j^L \prod_{j\in T_{2,a}}\mathrm{trnc}(z_j^L)\right]\right| = O_d\left(\frac{\mathbb{E}[(z_i^L - \mathrm{trnc}(z_i^L))^2]^{1/2}}{\sqrt{d}}\right).$$

Although this is enough to show that the difference

$$\frac{1}{\alpha n}\sum_{a\in E}\left(\mathbb{E}[f(\mathrm{trnc}(\boldsymbol{z}_{\partial a}^L))] - \mathbb{E}[f(\boldsymbol{z}_{\partial a}^L)]\right) = O_d(\tfrac{1}{\sqrt{d}}),$$

we want this quantity to be negligible even among terms of size $O_d(\frac{1}{\sqrt{d}})$. We show this as the timestep δ also goes to 0.

[18] For convenience, we use the notation $\mathrm{trnc}(\boldsymbol{z}^L)$ or $R(\boldsymbol{z}^L)$ to apply trnc or R entrywise on $\boldsymbol{z}^L$.

▶ **Proposition 41.** *Suppose Assumption 1 holds. Then for every integer $L \geq 1$, there is a C such that*

$$\frac{1}{n}\sum_{i\in V}\mathbb{E}[(z_i^L - \mathrm{trnc}(z_i^L))^2] \leq C\delta + o_d(1)\,.$$

Proof. Consider the pseudo-Lipschitz function $\psi(x) \stackrel{\text{def}}{=} (x - \mathrm{trnc}(x))^2$. By Theorem 29,

$$\mathbb{E}[\psi(z_i^L)] = \mathbb{E}[\psi(\widetilde{Z}_L^\delta)] + o_d(1)\,.$$

We consider the behavior as δ goes to zero. Since ψ is bounded by a quadratic function,

$$\mathbb{E}[\psi(\widetilde{Z}_L^\delta)] \leq \mathbb{E}[|\psi(Z_{\delta L})|] + \mathbb{E}[|\widetilde{Z}_L^\delta - Z_{\delta L}|^2]\,.$$

The first term is zero by Lemma 38. The statement follows by Proposition 36. ◀

We combine these statements with Theorem 37 to achieve our goal:

▶ **Theorem 42.** *Suppose Assumption 1 holds. Fix a factor a. Then for all $\epsilon > 0$, there exist $d_0 \in \mathbb{N}$ and $\delta_0 > 0$ such that for all $d \geq d_0$ and $0 < \delta \leq \delta_0$,*

$$\frac{1}{\alpha n}\sum_{a\in E}\mathbb{E}\left[f(R(\boldsymbol{z}_{\partial a}^L))\right] \geq \mathbb{E}[f] + \frac{\mathsf{ALG}_\xi - \epsilon}{\sqrt{\alpha}}\,.$$

We conclude this section by showing the value of the algorithm *concentrates* around its mean. This implies that the algorithm is not only good on average, but it finds a good solution with high probability as $n \to \infty$.

▶ **Proposition 43** (Generalization of [3, Lemma 4.10])**.** *Fix a integer $L \geq 1$. Then there exists a constant C such that for all $t \geq 0$,*

$$\mathbb{P}\left(\left|\frac{1}{\alpha n}\sum_{a\in E}\left(f(R(\boldsymbol{z}_{\partial a}^L)) - \mathbb{E}[f(R(\boldsymbol{z}_{\partial a}^L))]\right)\right| \geq t\right) \leq Ce^{-nt^2/(2C)}\,.$$

Proof. Partition the nodes of the graph G into $C = C(L)$ sets $\{W_1, \dots, W_C\}$ such that any two nodes $i, j \in W_q$ have distance at least $2L + 3$. Then the value $\sum_{a\in E}\left(f(R(\boldsymbol{z}_{\partial a}^L)) - \mathbb{E}[f]\right)$ can be split into C sums as in

$$\Delta \stackrel{\text{def}}{=} \sum_{a\in E}\left(f(R(\boldsymbol{z}_{\partial a}^L)) - \mathbb{E}[f]\right) = \frac{1}{r}\sum_{i\in V}R(z_i^L)\sum_{a\in\partial i}\mathsf{D}_{i;a}f(R(\boldsymbol{z}_{\partial a}^L)) = \frac{1}{r}\sum_{\iota=1}^{C}\Delta_\iota\,,$$

where $\Delta_\iota \stackrel{\text{def}}{=} \sum_{i\in W_\iota}R(z_i^L)\sum_{a\in\partial i}\mathsf{D}_{i;a}f(R(\boldsymbol{z}_{\partial a}^L))$. By a union bound, $\mathbb{P}((\Delta - \mathbb{E}[\Delta]) \geq t\alpha n) \leq \sum_{\iota=1}^{C}\mathbb{P}((\Delta_\iota - \mathbb{E}[\Delta_\iota]) \geq tdn/C)$. Since each Δ_ι is a sum of up to n independent variables each bounded by d, this probability is at most $C\exp\left(-\frac{(tdn/C)^2}{8nd^2}\right) = C\exp\left(-\frac{nt^2}{8C^2}\right)$. ◀

6 Open questions

The first challenge is to either prove Assumption 1, or mathematize the continuity argument that using a near-minimizer $\widetilde{\mu}_* \in \mathscr{L}_\xi$ in the algorithm incurs negligible error. This would address the chief concern of Corollary 7.

As mentioned before, our algorithm assumes the CSP predicate f has no linear terms; e.g. f cannot represent k-SAT. We expect that a preprocessing step fixes this issue, similar to [79]'s improvement to [1].

Current techniques obstructing overlap-concentrated algorithms [50, 55] are restricted to *even* Ising spin glasses and CSP predicates f. However, a recent technique [47] on spherical spin glasses seems to circumvent this issue. Applying this technique to Ising spin glasses could lead to obstructions beyond even models, which automatically transfer to $\mathsf{CSP}_f(\alpha)$ by [55]. We suspect that message-passing is optimal among *overlap-concentrated* algorithms for *all* Ising spin glasses and CSP predicates f.

Furthermore, this work deepens a curious connection between $\mathsf{CSP}_f(\alpha)$ and Ising spin glasses [26, 55]. For example, this algorithm and the one in [1] are very similar, and achieve the same performance up to affine shift. In fact, the same is true for the QAOA for any constant depth and choice of hyperparameters [11, 12]! (This is known for monomial predicates f but likely extends to all predicates.) Perhaps there is a way to characterize the equivalence in energy landscape that explains why some algorithms "can't see the difference" between the two models. This equivalence extends beyond overlap gap phenomena, which are defined only at near-optimal energies.

It seems that understanding optimal algorithms for $\mathsf{CSP}_f(\alpha)$ without asymptotics in α requires new techniques.

References

1 Ahmed El Alaoui, Andrea Montanari, and Mark Sellke. Optimization of mean-field spin glasses. *The Annals of Probability*, 49(6):2922–2960, 2021. arXiv:2001.00904.

2 Ahmed El Alaoui, Andrea Montanari, and Mark Sellke. Sampling from the sherrington-kirkpatrick gibbs measure via algorithmic stochastic localization. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 323–334. IEEE, 2022. doi:10.1109/FOCS54457.2022.00038.

3 Ahmed El Alaoui, Andrea Montanari, and Mark Sellke. Local algorithms for maximum cut and minimum bisection on locally treelike regular graphs of large degree, 2023. arXiv:2111.06813.

4 Anurag Anshu and Tony Metger. Concentration bounds for quantum states and limitations on the QAOA from polynomial approximations. *Quantum*, 7:999, May 2023. doi:10.22331/q-2023-05-11-999.

5 Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando G. S. L. Brandao, David A. Buell, Brian Burkett, Yu Chen, Zijun Chen, Ben Chiaro, Roberto Collins, William Courtney, Andrew Dunsworth, Edward Farhi, Brooks Foxen, Austin Fowler, Craig Gidney, Marissa Giustina, Rob Graff, Keith Guerin, Steve Habegger, Matthew P. Harrigan, Michael J. Hartmann, Alan Ho, Markus Hoffmann, Trent Huang, Travis S. Humble, Sergei V. Isakov, Evan Jeffrey, Zhang Jiang, Dvir Kafri, Kostyantyn Kechedzhi, Julian Kelly, Paul V. Klimov, Sergey Knysh, Alexander Korotkov, Fedor Kostritsa, David Landhuis, Mike Lindmark, Erik Lucero, Dmitry Lyakh, Salvatore Mandrà, Jarrod R. McClean, Matthew McEwen, Anthony Megrant, Xiao Mi, Kristel Michielsen, Masoud Mohseni, Josh Mutus, Ofer Naaman, Matthew Neeley, Charles Neill, Murphy Yuezhen Niu, Eric Ostby, Andre Petukhov, John C. Platt, Chris Quintana, Eleanor G. Rieffel, Pedram Roushan, Nicholas C. Rubin, Daniel Sank, Kevin J. Satzinger, Vadim Smelyanskiy, Kevin J. Sung, Matthew D. Trevithick, Amit Vainsencher, Benjamin Villalonga, Theodore White, Z. Jamie Yao, Ping Yeh, Adam Zalcman, Hartmut Neven, and John M. Martinis. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, October 2019. doi:10.1038/s41586-019-1666-5.

6 Antonio Auffinger and Wei-Kuo Chen. Parisi formula for the ground state energy in the mixed p-spin model. *The Annals of Probability*, 45(6B):4617–4631, 2017. arXiv:1606.05335.

7 Antonio Auffinger, Andrea Montanari, and Eliran Subag. Optimization of random high-dimensional functions: Structure and algorithms. In Patrick Charbonneau, Enzo Marinari, Giorgio Parisi, Federico Ricci-Tersenghi, Gabriele Sicuro, Francesco Zamponi, and Marc Mézard, editors, *Spin Glass Theory and Far Beyond: Replica Symmetry Breaking After 40 Years*, chapter 29. World Scientific, 2023. arXiv:2206.10217.

8 Boaz Barak and Kunal Marwaha. Classical algorithms and quantum limitations for maximum cut on high-girth graphs. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*, 2022. doi:10.4230/LIPIcs.ITCS.2022.14.

9 Boaz Barak, Ankur Moitra, Ryan O'Donnell, Prasad Raghavendra, Oded Regev, David Steurer, Luca Trevisan, Aravindan Vijayaraghavan, David Witmer, and John Wright. Beating the random assignment on constraint satisfaction problems of bounded degree. In *APPROX/RANDOM 2015*, pages 110–123, 2015. arXiv:1505.03424.

10 Aseem Baranwal, Kimon Fountoulakis, and Aukosh Jagannath. Optimality of message-passing architectures for sparse graphs. *Advances in Neural Information Processing Systems*, 36:40320–40341, 2023. doi:10.48550/arXiv.2305.10391.

11 Joao Basso, Edward Farhi, Kunal Marwaha, Benjamin Villalonga, and Leo Zhou. The Quantum Approximate Optimization Algorithm at high depth for MaxCut on large-girth regular graphs and the Sherrington-Kirkpatrick model. In *17th Conference on the Theory of Quantum Computation, Communication and Cryptography*, 2022. arXiv:2110.14206.

12 Joao Basso, David Gamarnik, Song Mei, and Leo Zhou. Performance and limitations of the QAOA at constant levels on large sparse hypergraphs and spin glass models. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, October 2022. doi:10.1109/focs54457.2022.00039.

13 Mohsen Bayati and Andrea Montanari. The dynamics of message passing on dense graphs, with applications to compressed sensing. *IEEE Transactions on Information Theory*, 57(2):764–785, 2011. arXiv:1001.3448.

14 Erwin Bolthausen. An iterative construction of solutions of the TAP equations for the Sherrington-Kirkpatrick model. *Communications in Mathematical Physics*, 325(1):333–366, 2014. arXiv:1201.2891.

15 Adam Bouland, Bill Fefferman, Chinmay Nirkhe, and Umesh Vazirani. On the complexity and verification of quantum random circuit sampling. *Nature Physics*, 15(2):159–163, October 2018. doi:10.1038/s41567-018-0318-2.

16 Sami Boulebnane, Abid Khan, Minzhao Liu, Jeffrey Larson, Dylan Herman, Ruslan Shaydulin, and Marco Pistoia. Evidence that the Quantum Approximate Optimization Algorithm optimizes the Sherrington-Kirkpatrick model efficiently in the average case, 2025. arXiv:2505.07929.

17 Sami Boulebnane and Ashley Montanaro. Predicting parameters for the Quantum Approximate Optimization Algorithm for MAX-CUT from the infinite-size limit, 2021. arXiv:2110.10685.

18 Anne Broadbent, Joseph Fitzsimons, and Elham Kashefi. Universal blind quantum computation. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, October 2009. doi:10.1109/focs.2009.36.

19 Houssam El Cheairi and David Gamarnik. Algorithmic universality, low-degree polynomials, and Max-Cut in sparse random graphs, 2024. doi:10.48550/arXiv.2412.18014.

20 Wei Kuo Chen. Variational representations for the Parisi functional and the two-dimensional Guerra-Talagrand bound. *Annals of Probability*, 45(6):3929–3966, 2017. arXiv:1501.06635.

21 Wei-Kuo Chen, David Gamarnik, Dmitry Panchenko, and Mustazee Rahman. Suboptimality of local algorithms for a class of max-cut problems. *The Annals of Probability*, 47(3):1587–1618, 2019. arXiv:1707.05386.

22 Wei-Kuo Chen and Dmitry Panchenko. On the TAP free energy in the mixed p-spin models. *Communications in Mathematical Physics*, 362:219–252, 2018. arXiv:1709.03468.

23 Wei-Kuo Chen, Dmitry Panchenko, and Eliran Subag. Generalized TAP free energy. *Communications on Pure and Applied Mathematics*, 76(7):1329–1415, 2023. arXiv:1812.05066.

24 Chi-Ning Chou, Peter J Love, Juspreet Singh Sandhu, and Jonathan Shi. Limitations of local quantum algorithms on random MAX-k-XOR and beyond. In *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, 2022. doi:10.4230/LIPIcs.ICALP.2022.41.

25 Alexander M Dalzell, Nicola Pancotti, Earl T Campbell, and Fernando GSL Brandão. Mind the gap: Achieving a super-Grover quantum speedup by jumping to the end. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1131–1144, 2023. arXiv:2212.01513.

26 Amir Dembo, Andrea Montanari, and Subhabrata Sen. Extremal cuts of sparse random graphs. *The Annals of Probability*, 45(2):1190–1217, March 2017. doi:10.1214/15-AOP1084.

27 Jian Ding, Allan Sly, and Nike Sun. Maximum independent sets on random regular graphs. *Acta Mathematica*, 217:263–340, 2016. arXiv:1310.4787.

28 Jian Ding, Allan Sly, and Nike Sun. Proof of the satisfiability conjecture for large k. *Annals of Mathematics*, 196(1):1–388, 2022. arXiv:1411.0650.

29 David L Donoho, Arian Maleki, and Andrea Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, 2009. arXiv:0907.3574.

30 Richard Durrett. *Probability: Theory and Examples*. Cambridge University Press, fifth edition, 2019. doi:10.1017/9781108591034.

31 Sepehr Ebadi, Alexander Keesling, Madelyn Cain, Tout T Wang, Harry Levine, Dolev Bluvstein, Giulia Semeghini, Ahmed Omran, J-G Liu, Rhine Samajdar, et al. Quantum optimization of maximum independent set using rydberg atom arrays. *Science*, 376(6598):1209–1215, 2022. arXiv:2202.09372.

32 Edward Farhi, David Gamarnik, and Sam Gutmann. The Quantum Approximate Optimization Algorithm needs to see the whole graph: A typical case, 2020. arXiv:2004.09002.

33 Edward Farhi, David Gamarnik, and Sam Gutmann. The Quantum Approximate Optimization Algorithm needs to see the whole graph: Worst case examples, 2020. arXiv:2005.08747.

34 Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A Quantum Approximate Optimization Algorithm, 2014. arXiv:1411.4028.

35 Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A Quantum Approximate Optimization Algorithm applied to a bounded occurrence constraint problem, 2015. arXiv:1412.6062.

36 Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Leo Zhou. The Quantum Approximate Optimization Algorithm and the Sherrington-Kirkpatrick model at infinite size. *Quantum*, 6:759, July 2022. doi:10.22331/q-2022-07-07-759.

37 Edward Farhi, Sam Gutmann, Daniel Ranard, and Benjamin Villalonga. Lower bounding the MaxCut of high girth 3-regular graphs using the QAOA, 2025. doi:10.48550/arXiv.2503.12789.

38 Oliver Y Feng, Ramji Venkataramanan, Cynthia Rush, and Richard J Samworth. A unifying tutorial on approximate message passing. *Foundations and Trends® in Machine Learning*, 15(4):335–536, 2022. arXiv:2105.02180.

39 Yaotian Fu and Philip W. Anderson. Application of statistical mechanics to NP-complete problems in combinatorial optimisation. *Journal of Physics A: Mathematical and General*, 19(9):1605–1620, June 1986. doi:10.1088/0305-4470/19/9/033.

40 David Gamarnik. The overlap gap property: A topological barrier to optimizing over random structures. *Proceedings of the National Academy of Sciences*, 118(41), October 2021. doi:10.1073/pnas.2108492118.

41 David Gamarnik. Barriers for the performance of graph neural networks (gnn) in discrete random structures. a comment, 2023. arXiv:2306.02555.

42 David Gamarnik, Aukosh Jagannath, and Alexander S. Wein. Circuit lower bounds for the p-spin optimization problem, 2021. arXiv:2109.01342.

43 David Gamarnik, Aukosh Jagannath, and Alexander S Wein. Hardness of random optimization problems for boolean circuits, low-degree polynomials, and Langevin dynamics. *SIAM Journal on Computing*, 53(1):1–46, 2024. arXiv:2004.12063.

44 David Gamarnik and Madhu Sudan. Limits of local algorithms over sparse random graphs. In *Proceedings of the 5th conference on Innovations in Theoretical Computer Science*, pages 369–376, 2014. arXiv:1304.1831.

45 M. B. Hastings. Classical and quantum bounded depth approximation algorithms, 2019. arXiv:1905.07047.

46 Brice Huang. *Computational Hardness in Random Optimization Problems from the Overlap Gap Property*. Master's thesis, Massachusetts Institute of Technology, 2022. URL: https://dspace.mit.edu/handle/1721.1/143164.

47 Brice Huang and Mark Sellke. Algorithmic threshold for multi-species spherical spin glasses, 2023. doi:10.48550/arXiv.2303.12172.

48 Brice Huang and Mark Sellke. Optimization algorithms for multi-species spherical spin glasses. *Journal of Statistical Physics*, 191(2):29, 2024. arXiv:2308.09672.

49 Brice Huang and Mark Sellke. Strong low degree hardness for stable local optima in spin glasses, 2025. doi:10.48550/arXiv.2501.06427.

50 Brice Huang and Mark Sellke. Tight Lipschitz hardness for optimizing mean field spin glasses. *Communications on Pure and Applied Mathematics*, 78(1):60–119, 2025. arXiv:2110.07847.

51 Neng Huang, Will Perkins, and Aaron Potechin. Hardness of sampling for the anti-ferromagnetic Ising model on random graphs, 2024. doi:10.48550/arXiv.2409.03974.

52 Misha Ivkov and Tselil Schramm. Semidefinite programs simulate approximate message passing robustly. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, pages 348–357, 2024. arXiv:2311.09017.

53 Aukosh Jagannath and Ian Tobasco. A dynamic programming approach to the Parisi functional. *Proceedings of the American Mathematical Society*, 144(7):3135–3150, December 2015. doi:10.1090/proc/12968.

54 David Jekel, Juspreet Singh Sandhu, and Jonathan Shi. Potential Hessian ascent: The Sherrington-Kirkpatrick model. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 5307–5387. SIAM, 2025. arXiv:2408.02360.

55 Chris Jones, Kunal Marwaha, Juspreet Singh Sandhu, and Jonathan Shi. Random Max-CSPs inherit algorithmic hardness from spin glasses. In *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, 2023. doi:10.4230/LIPIcs.ITCS.2023.77.

56 Stephen P Jordan, Noah Shutty, Mary Wootters, Adam Zalcman, Alexander Schmidhuber, Robbie King, Sergei V Isakov, Tanuj Khattar, and Ryan Babbush. Optimization by decoded quantum interferometry, 2024. arXiv:2408.08292.

57 Yoshiyuki Kabashima. A CDMA multiuser detection algorithm on the basis of belief propagation. *Journal of Physics A: Mathematical and General*, 36(43):11111, 2003. doi:10.1088/0305-4470/36/43/030.

58 Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT press, 2009. URL: https://dl.acm.org/doi/abs/10.5555/1795555.

59 Florent Krzakala, Marc Mézard, Francois Sausset, Yifan Sun, and Lenka Zdeborová. Probabilistic reconstruction in compressed sensing: Algorithms, phase diagrams, and threshold achieving matrices. *Journal of Statistical Mechanics: Theory and Experiment*, 2012(08):P08009, 2012. arXiv:1206.3953.

60 Frank R Kschischang, Brendan J Frey, and H-A Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001. doi:10.1109/18.910572.

61 Thibault Lesieur, Florent Krzakala, and Lenka Zdeborová. Constrained low-rank matrix estimation: Phase transitions, approximate message passing and applications. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(7):073403, 2017. arXiv:1701.00858.

62 Filip B Maciejewski, Stuart Hadfield, Benjamin Hall, Mark Hodson, Maxime Dupont, Bram Evert, James Sud, M Sohaib Alam, Zhihui Wang, Stephen Jeffrey, et al. Design and execution of quantum circuits using tens of superconducting qubits and thousands of gates for dense Ising optimization problems. *Physical Review Applied*, 22(4):044074, 2024. arXiv:2308.12423.

63 Kunal Marwaha. Local classical MAX-CUT algorithm outperforms p=2 QAOA on high-girth regular graphs. *Quantum*, 5:437, April 2021. doi:10.22331/q-2021-04-20-437.

64 Sohir Maskey, Ron Levie, Yunseok Lee, and Gitta Kutyniok. Generalization analysis of message passing neural networks on large random graphs. *Advances in neural information processing systems*, 35:4805–4817, 2022. arXiv:2202.00645.

65 Marc Mézard and Andrea Montanari. *Information, Physics, and Computation*. Oxford University Press, 2009. doi:10.1093/acprof:oso/9780198570837.001.0001.

66 Marc Mézard, Giorgio Parisi, and Miguel Angel Virasoro. *Spin Glass Theory and Beyond: An Introduction to the Replica Method and Its Applications*, volume 9. World Scientific Publishing Company, 1987. doi:10.1142/0271.

67 Marco Mondelli and Ramji Venkataramanan. Approximate message passing with spectral initialization for generalized linear models. In *International Conference on Artificial Intelligence and Statistics*, pages 397–405. PMLR, 2021. arXiv:2010.03460.

68 Andrea Montanari. Optimization of the Sherrington-Kirkpatrick Hamiltonian. *SIAM Journal on Computing*, 2021. arXiv:1812.10897.

69 Andrea Montanari, Yonina C. Eldar, and Gitta Kutyniok. Graphical models concepts in compressed sensing. *Compressed Sensing*, pages 394–438, 2012. arXiv:1011.4328.

70 Dmitry Panchenko. The Parisi ultrametricity conjecture. *Annals of Mathematics*, pages 383–393, 2013. arXiv:1112.1003.

71 Dmitry Panchenko. *The Sherrington-Kirkpatrick Model*. Springer Science & Business Media, 2013. doi:10.1007/978-1-4614-6289-7.

72 Dmitry Panchenko. On the k-sat model with large number of clauses. *Random Structures & Algorithms*, 52(3):536–542, 2018. arXiv:1608.06256.

73 Giorgio Parisi. Infinite number of order parameters for spin-glasses. *Phys. Rev. Lett.*, 43:1754–1756, December 1979. doi:10.1103/PhysRevLett.43.1754.

74 John Preskill. Quantum computing in the NISQ era and beyond. *Quantum*, 2:79, August 2018. doi:10.22331/q-2018-08-06-79.

75 Tom Richardson and Ruediger Urbanke. *Modern Coding Theory*. Cambridge University Press, 2008. doi:10.1017/CBO9780511791338.

76 Cynthia Rush, Adam Greig, and Ramji Venkataramanan. Capacity-achieving sparse superposition codes via approximate message passing decoding. *IEEE Transactions on Information Theory*, 63(3):1476–1500, 2017. arXiv:1501.05892.

77 Ciarán Ryan-Anderson. *Quantum Algorithms, Architecture, and Error Correction*. PhD thesis, The University of New Mexico, 2018. arXiv:1812.04735.

78 Juspreet Singh Sandhu and Jonathan Shi. Sum-of-squares & Gaussian processes I: Certification, 2024. arXiv:2401.14383.

79 Mark Sellke. Optimizing mean field spin glasses with external field. *Electronic Journal of Probability*, 29:1–47, 2024. arXiv:2105.03506.

80 Subhabrata Sen. Optimization on sparse random hypergraphs and spin glasses. *Random Structures & Algorithms*, 53(3):504–536, 2018. doi:10.1002/rsa.20774.

81 Ruslan Shaydulin and Marco Pistoia. Qaoa with $n \cdot p \geq 200$. In *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, volume 1, pages 1074–1077. IEEE, 2023. arXiv:2303.02064.

82 David Sherrington and Scott Kirkpatrick. Solvable model of a spin-glass. *Physical Review Letters*, 35(26):1792, 1975. doi:10.1103/PhysRevLett.35.1792.

83 Allan Sly and Nike Sun. The computational hardness of counting in two-spin models on d-regular graphs. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 361–369. IEEE, 2012. arXiv:1203.2602.

84 Eliran Subag. Following the ground states of full-RSB spherical spin glasses. *Communications on Pure and Applied Mathematics*, 74(5):1021–1044, 2021. arXiv:1812.04588.

85 Michel Talagrand. The Parisi formula. *Annals of Mathematics*, pages 221–263, 2006. URL: https://www.jstor.org/stable/20159953.

86 Ewin Tang. Dequantizing algorithms to understand quantum advantage in machine learning. *Nature Reviews Physics*, 4:692–693, 2022. doi:10.1038/s42254-022-00511-w.

87 David J Thouless, Philip W Anderson, and Robert G Palmer. Solution of 'solvable model of a spin glass'. *Philosophical Magazine*, 35(3):593–601, 1977. doi:10.1080/14786437708235992.

88 Zhihui Wang, Stuart Hadfield, Zhang Jiang, and Eleanor G. Rieffel. Quantum approximate optimization algorithm for MaxCut: A fermionic view. *Phys. Rev. A*, 97:022304, February 2018. doi:10.1103/PhysRevA.97.022304.

89 Jonathan Wurtz and Peter Love. MaxCut Quantum Approximate Optimization Algorithm performance guarantees for p>1. *Physical Review A*, 103(4), April 2021. doi:10.1103/physreva.103.042612.

90 Lenka Zdeborová and Florent Krzakala. Statistical physics of inference: Thresholds and algorithms. *Advances in Physics*, 65(5):453–552, 2016. arXiv:1511.02476.

91 Han-Sen Zhong, Hui Wang, Yu-Hao Deng, Ming-Cheng Chen, Li-Chao Peng, Yi-Han Luo, Jian Qin, Dian Wu, Xing Ding, Yi Hu, Peng Hu, Xiao-Yan Yang, Wei-Jun Zhang, Hao Li, Yuxuan Li, Xiao Jiang, Lin Gan, Guangwen Yang, Lixing You, Zhen Wang, Li Li, Nai-Le Liu, Chao-Yang Lu, and Jian-Wei Pan. Quantum computational advantage using photons. *Science*, 370(6523):1460–1463, 2020. doi:10.1126/science.abe8770.